STUDIES IN LOGIC

AND

THE FOUNDATIONS OF MATHEMATICS

NORTH-HOLLAND PUBLISHING COMPANY
AMSTERDAM

DICTIONARY OF SYMBOLS OF MATHEMATICAL LOGIC

edited by
ROBERT FEYS
and
FREDERIC B. FITCH

1969

NORTH-HOLLAND PUBLISHING COMPANY
AMSTERDAM

Library of Congress Catalog Card Number: 67-30883

PRINTED IN THE NETHERLANDS

CONTENTS

FOREWORD

This "Dictionary" of the symbols of mathematical logic was organized mainly under the direction of the late Professor Robert Feys. The Preface has been retained essentially as he wrote it and serves to indicate both his conception of the Dictionary and the nature of the contributions made to the first draft by many contributors, especially Professor John R. Myhill.

The present version results from an extensive and fundamental rewriting of the earlier draft in order to improve clarity of exposition and increase, to some degree, uniformity of conventions as among parts written by different authors. Specific page references to other works were largely lacking in the first draft and have been inserted. Numerous examples, such as examples of axiom sets, have been added. Much new material has been added, and except for parts of the last chapter, practically every sentence of the original draft has been reformulated or superseded.

I am under no illusions that the result will please all readers, or that the title of "Dictionary" is altogether appropriate. It was at the request of Professor S.C. Kleene, then President of the Association for Symbolic Logic, that I undertook the task of producing the present revision. I had nothing to do with the earlier phases of the planning and writing of the Dictionary, but I hope that those earlier efforts can now be regarded as brought to a successful conclusion.

Whatever the deficiencies of the present version may be, I must myself accept main responsibility for them. They might have been less had my obligations in other directions been fewer.

Without the advice and help offered so unstintingly by Professor Patricia James, this version of the Dictionary could not have been written, and I am extremely grateful to her. I also wish to express gratitude to Professor James William Dickoff and Professor Alan R. Anderson for their help and valuable suggestions. Thanks are also due for partial fi-

nancial assistance made available by the Bell Telephone Laboratories of Murray Hill, New Jersey. Professor Alonzo Church made many useful suggestions and criticisms that were taken into account in the final stage of revision, and I express my gratitude to him also.

Frederic B. Fitch

PREFACE
OF THE ORIGINAL EDITOR

We shall not begin with a definition of "mathematical logic" or "symbolic logic". We content ourselves with trying to give a translation for the symbols usually considered as the symbols of mathematical logic.

The purpose of the present "Dictionary" is to enable the reader to find with some ease the meaning and interpretation of symbols currently used in mathematical logic (symbolic logic). The criterion for "current use" in mathematical logic is as follows. The Dictionary includes (1) symbols used in the recent handbooks mentioned in the list of abbreviations below, (2) symbols used in many of the articles in *The Journal of Symbolic Logic* from 1950 onwards, and (3) some notations that have been taken over from other sources and papers.

This Dictionary is intended for readers not having previous knowledge of mathematical logic as well as for logicians who want an explanation of a notation outside their usual fields. The non-logician might hope to find here a direct verbal translation for each symbol, just as in a usual dictionary. However, it would be misleading, especially for him, if we confined ourselves to what is a tentative translation. Such a procedure might confirm this non-logician (or non-mathematical-logician) in the widespread error that logical symbols are simply a sort of shorthand for ideas, the exact content of which can be grasped intuitively. It might thereby lead him to overlook the essential fact that the language of mathematical logic is not like other languages, but is rather the language of a *formalized* theory.

That it is a formalized theory signifies that the deductions can be made without any reference to what the meaning of the symbols of the system might be. It also signifies that the possible meanings of the symbols are only *implicit* in the axioms (primitive theorems), rules, and definitions of the system, and that they can be made explicit only to some extent by referring back to the axioms, rules, and definitions. It is for this reason

that the present Dictionary consists of a *systematic explanation* of the meaning of each symbol, discussed in the context of the corresponding formalized system. The references in the *Index of Symbols* are to sections of this systematic explanation. The discussions found at the indicated sections will provide the reader with tentative verbal translations and with hints for avoiding possible misunderstandings. The explanations never pretend to give a definitive solution of the problem of meaning but only the elements of such a solution.

The symbols of metamathematics in Chapter 9 are the only ones used in an intuitive sense and as having a meaning which does not need to be inferred from an axiomatic system. But even in this case an explanation, and not a bare translation, is what is most needed, as the intended meaning may be complex and technical.

A last difficulty should be stressed. The notation of recent mathematical logic is far from uniform. Not only do we have to take account of a very great variety of formalized systems with similar notation, but we must recognize the fact that various writers frequently express the same idea by means of different symbols and that the same symbol is employed in various senses by various authors. This is not, of course, a very satisfactory state of affairs, but it was not incumbent on us to express a preference for any one notation over another, nor do we discuss the possibility of a unified notation.

Such a situation calls for a last practical indication concerning the use of this Dictionary. The symbolic vocabulary of mathematical logic – especially the most usual one – is not very extended. But we have to do with various symbolisms and not just one, and even with more or less overlapping ideographic alphabets. This makes it impossible for a user of the Dictionary even to *find* a translation by some mechanical process such as the use of the lexicographic order. Even for "classical" texts the user will need to have a previous idea of the kind of notation used in the text to be translated (this may be suggested by among other things the nationality of the writer), and it may also be useful to take account of the kind of logic under consideration.

The publication of this Dictionary was undertaken as the suggestion of Mr. Paul Carus, and the cost of printing was borne by the Edward C. Hegeler Trust Fund, for which we express here our appreciation and thanks. We are also grateful for the help provided by the Belgian Centre de Recherches de Logique.

The preparatory steps toward the composition of the Dictionary were made by a committee formed of Professors Barlaz, Beth (Chairman), Church, Myhill, Quine, and Scholz. The present [1957] editing committee consists of Professors Barlaz, Beth, Church, Feys (Chairman), and Myhill.

This Dictionary is in great part the work of Professor Myhill, of Berkeley, who laid down its plan and wrote the fundamental Chapters 1 and 9 in their entirety. Professor Hailperin wrote a first draft for Chapters 2 and 3. We wish to thank Professor E. W. Beth, Professor W. T. Parry, Professor J. Dopp, and Dr. Ladrière for their careful revisions of the present text, and Mrs. De Permentier and Miss Chavasse for their help in preparing it.

Robert Feys

ABBREVIATIONS

The following abbreviations are used throughout this Dictionary:

[CF] for Curry and Feys, *Combinatory Logic*
[Ch] for Church, *Mathematical Logic*
[F] for Fitch, *Symbolic Logic*
[HA] for Hilbert and Ackermann, *Principles of Mathematical Logic*
[HB] for Hilbert and Bernays, *Grundlagen der Mathematik* – i, ii
[K] for Kleene, *Introduction to Metamathematics*
[PM] for Russell and Whitehead, *Principia Mathematica*
[Q] for Quine, *Mathematical Logic*
[R] for Rosser, *Logic for Mathematicians.*

Other books are referred to as usually done in *The Journal of Symbolic Logic*. Thus, 1085 will refer to the entry so numbered in the Bibliography to be found in Volume 1, pp. 121–218 and Volume 3, pp. 178–212 in the *Journal*. II 37 will refer to the book or paper whose review begins on page 37 of Volume II of the *Journal*. When the work referred to is a book, page numbers will be included; if the work is an article, no page references are given.

PRELIMINARIES

00. Translation and meaning

00.1. The usual purpose of a dictionary written in a language L and dealing with a language L' is to give a *translation* of the words of L' by means of the words of L. Words may be grouped, and some of these groupings will be sentences. A language may be considered to be a collection of sentences, each of which is formed by joining words together and using punctuation. The language of symbolic logic uses special symbols instead of ordinary words, and the sentences of symbolic logic are formed by joining these symbols together in accordance with specific principles or grammatical rules.

00.2. To explain the meanings of these sentences of symbolic logic it might seem sufficient to assign to each symbol an English word which would be its translation. Thus if L is English and L' is the language of symbolic logic, then each symbol of L' would be translated into a word of L. But some knowledge of the grammar of L' is also very important in explaining the meanings of the sentences of L'. Even if L' were an ordinary language, rather than a symbolic language, some knowledge of the grammar of L' would be required, including rules of declension and conjugation, syntactical rules of sentence formation, and rules concerned with context. The statement of these rules is necessary because the translation of a given expression depends on the fact that some ideas will be expressed not by the mere addition of words, but rather by the order of words and by use of punctuation and context. The statement of such rules is omitted in most dictionaries because the dictionaries are supposed to be used in conjunction with appropriate grammars and because the grammatical rules for L and L' are often similar. The present Dictionary, however, must largely supply its own grammatical information, and its

ABBREVIATIONS

The following abbreviations are used throughout this Dictionary:

[CF] for Curry and Feys, *Combinatory Logic*
[Ch] for Church, *Mathematical Logic*
[F] for Fitch, *Symbolic Logic*
[HA] for Hilbert and Ackermann, *Principles of Mathematical Logic*
[HB] for Hilbert and Bernays, *Grundlagen der Mathematik* – i, ii
[K] for Kleene, *Introduction to Metamathematics*
[PM] for Russell and Whitehead, *Principia Mathematica*
[Q] for Quine, *Mathematical Logic*
[R] for Rosser, *Logic for Mathematicians.*

Other books are referred to as usually done in *The Journal of Symbolic Logic*. Thus, 1085 will refer to the entry so numbered in the Bibliography to be found in Volume 1, pp. 121–218 and Volume 3, pp. 178–212 in the *Journal*. II 37 will refer to the book or paper whose review begins on page 37 of Volume II of the *Journal*. When the work referred to is a book, page numbers will be included; if the work is an article, no page references are given.

PRELIMINARIES

00. Translation and meaning

00.1. The usual purpose of a dictionary written in a language L and dealing with a language L' is to give a *translation* of the words of L' by means of the words of L. Words may be grouped, and some of these groupings will be sentences. A language may be considered to be a collection of sentences, each of which is formed by joining words together and using punctuation. The language of symbolic logic uses special symbols instead of ordinary words, and the sentences of symbolic logic are formed by joining these symbols together in accordance with specific principles or grammatical rules.

00.2. To explain the meanings of these sentences of symbolic logic it might seem sufficient to assign to each symbol an English word which would be its translation. Thus if L is English and L' is the language of symbolic logic, then each symbol of L' would be translated into a word of L. But some knowledge of the grammar of L' is also very important in explaining the meanings of the sentences of L'. Even if L' were an ordinary language, rather than a symbolic language, some knowledge of the grammar of L' would be required, including rules of declension and conjugation, syntactical rules of sentence formation, and rules concerned with context. The statement of these rules is necessary because the translation of a given expression depends on the fact that some ideas will be expressed not by the mere addition of words, but rather by the order of words and by use of punctuation and context. The statement of such rules is omitted in most dictionaries because the dictionaries are supposed to be used in conjunction with appropriate grammars and because the grammatical rules for L and L' are often similar. The present Dictionary, however, must largely supply its own grammatical information, and its

purpose is to aid in translating symbolic texts written in a formal language whose grammar differs greatly from the grammar of ordinary spoken language.

00.3. An explanatory dictionary attempts to explain the meanings of words or symbols. This Dictionary is intended to be a dictionary of both translation and explanation. Explanations of meanings are in fact required if a dictionary is to provide more than rather rough translations, since words or symbols of a given language L' seldom have such literal equivalents in a translating language L that these equivalents may simply be substituted for the corresponding words or symbols of L' in a word-for-word translation. This is especially true if the translated language, L', is the language of symbolic logic, as it is in this Dictionary. These explanations of meanings will involve discussions of the grammatical structure of the language of symbolic logic.

00.4. This Dictionary differs, also, from the usual dictionary in the fact that it is concerned with many languages and not just one language. In one sense, there is really no *one* language of symbolic logic, but rather many languages of symbolic logic. Each formalized system of logic is a language in itself. This fact must be kept in mind even if we do not constantly point out fine differences in meaning between symbols of different systems. It should also be kept in mind that some of the general explanations given in these introductory sections may not fit some special systems. This Dictionary, indeed, does not at all attempt to be exhaustive either with respect to the kinds of logical systems considered or the kinds of symbols considered. It attempts only to deal with some of the better known logical systems and logical notations.

01. Formalization

01.1. This Dictionary is concerned exclusively with the *formalized languages* of various systems of symbolic logic, that is, with *formalized deductive systems*. For each formalized language we must be able to specify which symbols are to be used, how they are to be combined to form formulas, and which formulas are to be regarded as provable in that language.

01.11. More precisely, a formalized language is defined by means of *formation rules* and *transformation rules*. The formation rules specify (1) the class of *primitive symbols* of the language, that is, the symbols of the language that are not compounded from other symbols of the language, and (2) the class of *compound expressions* which are systematically built from the primitive symbols in accordance with explicitly described principles of construction. Usually the compound expressions (together with certain of the primitive symbols) are referred to as the *well-formed formulas* (or simply, the *formulas*) of the language. The term "well-formed" is often abbreviated as "wf", and the term "well-formed formula" is often abbreviated as "wff". Among the wffs of a language there will usually be some, and sometimes all, that are treated as being capable of assertion. These are the *sentences*. Besides specifying which expressions are wf and which are not, the formation rules will often also provide some classification of the wffs. In particular, if there is to be a distinction between sentences and other wffs, the formation rules must provide this.

01.12. In addition to the formation rules, transformation rules are also required. The transformation rules are used to specify (1) when a sentence is a *consequence* of one or more other sentences, and (2) when a sentence is a *theorem*. (As suggested by Carnap, it is possible to subsume (2) under (1) if we allow ourselves to speak of consequences of the empty class of sentences.) A special case under (1) is concerned with a special kind of consequence called *direct consequence*. A special case under (2) is concerned with a special kind of theorems called *axioms*. (If Carnap's suggestion is followed, then axioms may be viewed as direct consequences of the empty class of sentences.) Rules specifying when a sentence is a direct consequence of one or more other sentences may also be called *underived rules of inference*. These are the rules that are, in a primary sense, used to derive theorems from axioms and from other theorems. The commonest way of specifying the class of theorems is as follows: First the axioms are specified and are considered to be theorems in a trivial sense. Then every sentence that is a direct consequence of other theorems by any one of the underived rules of inference is specified as also being a theorem. In a system that employs axioms, the usual situation is that some theorems are axioms, others are direct consequences of

axioms, while still others are consequences of axioms but not direct consequences of axioms.

01.13. Neither the formation rules nor the transformation rules depend on or involve any interpretation of the expressions of the language. These rules are concerned solely with the defining of classes of expressions (such as the class of wffs and the class of theorems) or important relations among expressions, but quite independently of what meaning may eventually be assigned to the expressions in question. In other words they are concerned with the *syntax* (or formal structure) of the language, and not at all with the *semantics* of the language (i.e., the analysis of the relation between the expressions of the language and their meaning). Nevertheless, if we are to have a language in the proper sense, rather than simply a *logistic system* or *calculus*, there must be an assignment of *meanings*, in some sense of that word; and any discussion of this is greatly facilitated by a knowledge of the formation and transformation rules of the language, that is, by a knowledge of the grammar or structure of the language.

01.14. The actual assignment of meanings to the wffs of a formalized language (and hence to the language itself) may be accomplished by informal discussions and explanations, as is done in this Dictionary, or else, in certain fairly simple or ideal cases, by what Carnap (XIV 237) has called *semantical rules*. Thus, on the one hand, the formation and transformation rules provide the syntax of a formalized language, while, on the other hand, informal interpretative discussions, or perhaps formal semantical rules, provide the semantics of the formalized language. If no semantics is provided, the formation and transformation rules give rise merely to an uninterpreted system of symbols and expressions. Such a system may be called an (uninterpreted) logistic system, or an (uninterpreted) calculus. If two or more interpretations are provided for such a calculus, two or more languages are thereby defined, but with exactly the same syntax. The membership of the class of wffs and the membership of the class of theorems are not in any way changed by this process of interpretation or assignment of meaning, because such membership is antecedently fixed by the formation and transformation rules.

02. Meaning, denotation and sense

02.1. In dealing with meanings of formulas of formalized languages, the theory of Frege (49*8*) and Church ([Ch] 4 ff.) is often useful. This theory may be expressed roughly as follows: A word, phrase or formula may be the name of something, and if it is, then whatever it is the name of is said to be its *denotation*, and it is said to denote that thing. Furthermore, a word, phrase or formula which is not the name of anything may still be meaningful and understandable and so may be said to have a *sense*. Thus the phrase, 'the solid gold mountain' has no denotation. But it does have a sense. A fairy tale containing the sentence, 'The prince after many days of riding came at last to the solid gold mountain', may well be perfectly intelligible as to sense. And yet this sentence, because of its use of the denotationless name, will either be factually false or fall into a category of neither true nor false (we need not here try to decide which). Frege's terms in German are 'Sinn' for sense and either 'Bedeutung' or 'Bezeichnung' for denotation. These have received a variety of different translations into English. 'Sense' has generally been used for 'Sinn'; but Russell (11*16*) uses 'meaning'. Other words used in place of 'denotation' are 'designatum' (which is the literal Latinization of 'Bezeichnung') and, by some recent writers, 'reference'. In summary, this view maintains that a word, phrase or formula may have both denotation and sense, or may have sense without denotation, though Frege recommended that, as a practical matter, denotationless names are to be avoided in the construction of a formalized language. (Not only the argument that there appear to be denotationless names, but also the following argument is sometimes offered for the Frege-Church theory: Two phrases having apparently different meaning may nevertheless denote the same thing, and it is hard to see how this can be the case unless we distinguish between sense and denotation. For example, the phrase, 'the Morning Star' and the phrase, 'the Evening Star' both denote the planet Venus but seem to have different meanings. These various arguments, though very persuasive, need not be regarded as ruling out an alternative position held by Russell.)

02.2. A method due to Russell (11*19*, and 26.43 below), known as *Russell's theory of descriptions*, emphasizes the notion of *contextual defi-*

nition and makes possible, in most cases, an account of meaning that dispenses with sense in favor of denotation, assuming, at least, that there are words, phrases or formulas to *denote* such *properties* as the property of being a solid gold mountain and the property of being the Morning Star.

03. Constants and variables

03.1. Unformalized languages use *proper names*. Formalized languages may do so also. In the case of formalized languages, these proper names are symbols or formulas and are called *constants*. When a formalized language is interpreted, it is usual to specify the denotations of the constants. Among the things that may be denoted by constants are classes, properties, relations, functions, propositions, individuals, and so on.

03.2. Many formalized languages make considerable use of *variables*, though some systems of logic avoid variables altogether (cp. 42.2, 45.6). Variables, like constants, may be viewed as names, but as names in a somewhat different way. A variable denotes or names *indeterminately* any one of *several* things which are called the *values* of the variable and which collectively comprise the *range* of the variable. Thus some class of individuals might constitute the range of a variable, and some class of classes (or some class of properties, relations, functions, or propositions) might constitute the range of another variable. In interpreting a formalized language it is usual to specify the respective ranges of the various variables of the language, just as it is usual to specify the respective things denoted by the various constants of the language.

04. Propositions

04.1. In addition to constants and variables, described above, symbolic logic also makes use of expressions which may be called *sentences* or *formal sentences* (cp. 01.11) and which may be thought of as expressing *propositions*. One way to fit sentences into a theory of meaning is to follow Frege's (498) and Church's ([Ch] 23 ff.) account of them as names of *truth-values*.

04.2. A declarative sentence of an unformalized language can, ordi rily, be supposed to enunciate some truth or falsehood, that is, some t proposition or some false proposition. In many formalized languag also, the same situation holds. It is usually agreed, therefore, tha sentence of such a formalized language has one of the *values* (*tr values*), *truth* or *falsehood*. In English grammatical usage a sentenc not a name of anything. Similarly, in the case of many formalized l guages, sentences are viewed as not being names but rather as *expres propositions* without serving as names of the propositions they expr On the Frege-Church theory, also, sentences are treated as not be names of propositions. On the contrary, they are treated as being names of truth-values of sentences. The *sense* of a sentence, on theory, is said to be the same thing as the proposition expressed by sentence. Thus, on the Frege-Church theory, a sentence has a proposit as its sense, and it has its truth-value (truth or falsehood) as its de tation. Just as we speak of sentences as having truth-values, so we r also (in a slightly different sense of 'having') speak of propositions having truth-values. More specifically, a proposition will have whiche truth-value any sentence expressing that proposition has.

04.3. On the Frege-Church theory, accordingly, a sentence denotes a name of) one of the truth-values truth and falsehood, and it conn (has as its sense) a proposition. Sentences having the same denotat may, on this theory, differ in connotation. Both of the sentences, '2 = and 'The earth is a planet', are true, and so denote truth, but they di in sense, that is, they express different propositions. (Notice the simila of this example to the example of phrases 'the Morning Star' and Evening Star' near the end of 02.1.)

04.4. As one alternative to the Frege-Church theory, however, i possible to regard sentences as names of propositions rather than as na of truth-values, and to treat truth-values simply as properties of pro sitions. Thus the truth-value truth would be viewed as a property p sessed by all true propositions, while the truth-value falsehood woulc viewed as a property possessed by all false propositions. The fact t English grammar generally fails to treat sentences as substantives does necessarily militate against this position in the case of formalized

guages. At least one widely used natural language, Chinese, does seem to use sentences as substantives.

04 5. As another alternative to the Frege-Church theory, we can treat sentences as names of propositions, as in the previous alternative, but treat truth-values as propositions rather than as properties of propositions. It is necessary to suppose, first, that some concept of *equivalence* among propositions has already been defined. Corresponding to every proposition p there is, we assume, an equivalent proposition, $*p$, which is called the *truth-value* of p. Thus every proposition is equivalent to (but not necessarily identical with) its truth-value. There is at most one truth-value for a given proposition, and every truth-value is itself a proposition. If two propositions p and q are equivalent, they will be assumed to have the same truth-value, so that $*p=*q$. If the kind of equivalence we are using is the very broad kind called *material equivalence* (10.46), then any two true propositions will be equivalent to each other in this broad sense, and any two false propositions will also be equivalent to each other. Hence all true propositions, being equivalent to one another, will have the same truth-value, called *truth*, and all false propositions, being equivalent to one another, will have the same truth-value, called *falsehood*. But truth and falsehood will themselves be propositions.

05. Forms and functions

05.1. First let us distinguish between *proper* and *improper* expressions. Proper expressions, such as names, have meaning in isolation, but improper expressions, such as parentheses and some connectives, do not have meaning in isolation. *Compound* expressions are built up usually by combining both proper and improper expressions, and they may themselves be either proper or improper. In the simplest case one or more constants are combined with parentheses and connectives ([Ch] 32) according to certain rules to give compound proper expressions. For example if we let the letter 'a' denote the class of rational beings and the letter 'b' denote the class of animals, we might use the compound expression '$a \cap b$' to denote the class of beings that belong to both a and b, that is, to denote the class of rational animals. (The single quotation marks are not part of the notation referred to. They merely indicate

that we are talking about the expressions that are placed between them; see 08.)

05.2. If in a compound expression we replace a constant by a variable, and if the replaced constant denotes something in the range of the variable, then the resulting compound expression may be called a *form* ([Ch] 10). In a similar way, several constants may be replaced by variables to produce a form. Suppose, for example, that 'u' and 'v' are being used as variables of a suitable kind, and that in the expression, 'animal and rational', we replace the nouns 'animal' and 'rational' respectively by 'u' and 'v' so as to give the expression, 'u and v'. Then the expression 'u and v' is a form. In a similar way we can obtain the form '$u \cap v$' from the expression '$a \cap b$', assuming that 'a' and 'b' are constants denoting things that fall respectively within the ranges of 'u' and 'v'.

05.3. Formalized logic deals to a great extent with *functions*, examples of which are the usual functions found in mathematics. We must distinguish between functions of *one argument*, functions of *two arguments*, functions of *three arguments*, and so on. We begin by considering functions of one argument. Associated with each function of one argument there is an *argument range* for that function (also frequently referred to as the *domain of definition* of the function). If a belongs to the argument range of a function f, and so can serve as an argument for f, then f may be applied to a so as to produce $f(a)$, which is called the *value* of the function f for the argument a. For example, the number 0 belongs to the argument range of the cosine function, so we may apply this function to the argument 0 to produce cos(0), which is 1; and therefore 1 is the value of the cosine function for the argument 0, that is, $\cos(0)=1$. If 'x' is being used as a variable, then the notation '$\cos(x)$' (or '$\cos x$') is sometimes used as a name of the cosine function, and '$\sin(x)$' (or '$\sin x$') is sometimes used as a name of the sine function, and so on, though strictly speaking 'cos' and 'sin' are correct abbreviated names of these functions. The expressions '$\cos(x)$' and '$\sin(x)$' should rather be considered to be *forms* (05.2) associated respectively with these two functions, instead of names for them. In general, if a variable has as its range the argument range of some function, we may write that variable in the argument position after the name of the function, and the result is a form associated with the function.

Thus if 'f' is being used as a name of some function and if 'x' is being used as a variable whose range is the same as the argument range of that function, then the expression '$f(x)$' is a form associated with that function. It should also be remarked that the class of all the values that a function takes for the various arguments in its argument range may be called the *value range* (or *range of values*) of the function.

05.4. A form associated with a function in the way described above may be called an *associated form* of the function. There is, indeed, a slightly more general way of constructing associated forms of functions. The method can easily be explained by way of an illustration. Suppose that the expression '$f(x)$' is an associated form of a function f in the sense already described, where 'x' is being used as a variable whose range is the argument range of f. Suppose also that the following equation is true whenever 'x' is replaced throughout it by a name of a member of the range of 'x':

$$f(x) = x^2 + 9x + 20.$$

In other words, suppose that whenever 'x' is replaced by a name of one of its values in the expression,

$$x^2 + 9x + 20,$$

the result is a name of the value of f for that value of the variable chosen as an argument. Then we may regard not only the expression '$f(x)$' as an associated form of the function f, but we may also regard the expression '$x^2+9x+20$' as an associated form of the function f. Thus there may be numerous expressions that are associated forms of a given function, but they are all in an important sense equivalent to one another.

05.5. In the case of functions of two arguments we must apply the function to an ordered pair of arguments. Thus if f is a function of two arguments, we say that $f(a, b)$ is the value of f for the ordered pair of arguments, a, b. The *argument range* (or *domain of definition*) of a function of two arguments is therefore a class of ordered pairs, consisting of every ordered pair to which the function may be applied so as to produce a value of the function. The first members of such ordered pairs may be

said to constitute the *first secondary argument range* of such a function, while the second members of such ordered pairs may be said to constitute the *second secondary argument range* of such a function. An associated form of a function of two arguments requires two variables, the first of which has as its range the first secondary argument range of the function, and the second of which has as range the second secondary argument range of the function. Functions of three or more arguments may be treated in a similar way. Thus the argument range of a function of three variables is a class of ordered triples; and, in general, a function of n arguments (for $n=2, 3, 4, \ldots$) has a class of ordered n-tuples as its argument range. All the first members of these n-tuples constitute the first secondary argument range of the function, all the second members of these n-tuples constitute the second secondary argument range of the function, and, in general, all the kth members of these n-tuples constitute the kth secondary argument range of the function. An associated form of a function of three variables must contain three variables, the ranges of which are respectively the three secondary argument ranges of the function. The variables, furthermore, must have some preassigned order, such as ordinary alphabetic order, so that the variable first in the (alphabetic) order, though perhaps not the leftmost variable of the associated form, will be understood to correspond to the first argument of the function, and the variable second in the (alphabetic) order will be understood to correspond to the second argument of the function, and so on. Consider, for example, the relatively simple case of functions of two arguments. Suppose that f and g are functions of two arguments and that the following equations hold for all numbers a and b in the respective first and second secondary argument ranges of the two functions:

$$f(a, b) = 2 + a^2 b,$$
$$g(a, b) = 2 + b^2 a.$$

Then, if our variables are ordered alphabetically, the expression '$2+x^2y$' would be an associated form of f but not of g, while the expression '$2+z^2w$' would be an associated form of g but not of f. We are assuming that 'w', 'x', 'y' and 'z' are here being used as variables whose ranges are respectively the first secondary argument range of g, the first secondary argument range of f, the second secondary argument range of f, and the second secondary argument range of g.

05.6. The *principle of extensionality for functions* asserts that if two functions have the same argument range and have the same value for each member of that range, then they are the same function. This principle holds in most classical systems of mathematical logic, but there are also systems of interest for which it fails, at least in its most general version.

05.7. *Properties* may be regarded as functions of one argument which have propositions as values. Thus properties have sometimes been referred to as *propositional functions*. More specifically, for example, any property of individuals may be viewed as a one-argument function whose argument range is a class of individuals and whose value range is a class of propositions. (Here by *individuals* we mean such things as physical objects, events, persons, and so on, and roughly speaking, anything other than propositions and functions. Sometimes, however, the term *individual* is used in a still broader way, as in 20.1.) Let 'f' denote the property blue, and let 'a' denote the sky. Then the sentence '$f(a)$' would express the proposition that the sky is blue. This proposition, $f(a)$, would be the value of the function (and in this case, property) blue when the sky is used as the individual which serves as argument. Furthermore, f (the property blue) is a *propositional* function, since its values are propositions. Suppose, now, that 'x' is used as a variable whose range is the class of all individuals of which we might conceivably predicate various colors. Then the range of 'x' could be regarded as the same as the argument range of the propositional function (or property) blue, and the expression '$f(x)$' could be regarded as a form associated with this same propositional function. Such a form, associated with a propositional function, may be called a *propositional form*. Notice that propositional forms are ordinarily obtainable from sentences simply by replacing constants by suitable variables. (Sometimes the term *propositional function* has been used to refer to the things that are here called propositional forms. Such terminology is avoided here as likely to lead to confusion.) The term *attribute* is often used as synonymous with the term *property*. In addition to properties of individuals, it is usually assumed that there are also properties of functions and properties of propositions. Thus properties themselves (that is, propositional functions) are assumed to have properties. The property of being a property is, for example, a property that all properties have. The property truth is a property that some propositions have and others do not have.

05.8. *Classes* (or *sets*) may be viewed as properties of a special kind, just as truth-values may be viewed as propositions of a special kind (04.5). On this view there corresponds to each property f a special property called the *extension of f* or *the class defined by f*. Thus, for example, the class of blue things is the extension of the property blue. In other words, it is the class defined by the property blue. Classes differ from properties of other sorts in being such that if two of them have exactly the same members (that is, are properties of exactly the same things), then those two classes are identical with each other, and hence are the same class and not really two. This latter principle is called the *principle of extensionality for classes*. It is derivable from the principle of extensionality for functions (05.6) and the account of truth-values given in 04.5, provided that by equivalence we mean material equivalence (10.46) and that by classes we mean one-argument propositional functions (properties) that take as their values only those special propositions that are truth-values. On the other hand, the essential difference between class and property is expressed in the Frege-Church theory of meaning not by saying that a class is a property of a special kind but by saying that a class is the denotation of whatever has a defining property of the class as its sense or connotation, so that classes and truth-values are denotations while properties and propositions are senses or connotations. For a more accurate and adequate account of the Frege-Church theory, however, the reader is advised to refer to the writings of Frege and Church.

05.9. *Relationships* (or *relations in intension*) may be regarded as propositional functions of two or more arguments. Thus an n-termed relationship is a propositional function of n arguments, where $n>1$. Those relationships all of whose values are truth-values, will be called simply *relations* (or *relations in extension*), just as properties all of whose values are truth-values are called classes. Thus, in the present terminology, relations correspond to relationships just as classes correspond to properties and just as truth-values correspond to propositions. (Indeed, we may view classes as one-termed relations and properties as one-termed relationships. Furthermore, we may view truth-values as zero-termed relations and propositions as zero-termed relationships.) If two relations relate exactly the same things in exactly the same way, then those relations are identical with each other and so are one and the same relation. This

principle is called the *principle of extensionality for relations*. It is derivable in essentially the same way that the principle of extensionality for classes is derivable. Relations can often conveniently be regarded as being classes of ordered couples (ordered pairs) in the case of two-termed relations (60.1), or as classes of ordered n-tuples in the case of n-termed relations. In this whole matter of the nature of functions, classes, and relations, the viewpoint of Bertrand Russell is partially different from the one outlined here. The only functions he accepts as logically basic are propositional functions. He treats classes and relations in terms of propositional functions by way of contextual definitions (50.4, 60.1, [PM] *20, *21). Functions other than propositional functions are treated by Russell as if they were relations of a special sort (65, [PM] *30).

06. Operators and bound variables

06.1. Certain expressions are called *operators*. An operator may be thought of as a name of an abstract entity called an *operation*. An operator may also be thought of simply as an expression which *operates on* another expression by being placed to the left or right of it. Thus the expression '$\sim$' is often used as an operator which serves as a name of the operation negation. This latter operation converts a proposition into its negate (or denial). We may also think of '$\sim$' as an operator in the sense that if it is placed to the left of a sentence it operates on the sentence to give the negate of that sentence. (The proposition is the meaning of a sentence, so that in one case a meaning is operated on to give a meaning, while in the other case a sentence is operated on to give a sentence.) Some operators contain one or more variables called *binding variables* of the operator, but other operators, such as '$\sim$', lack binding variables, and these operators are sometimes called *connectives*. An example of an operator that contains a binding variable is the existential quantifier, '$(\exists x)$', assuming that the symbol 'x' is being employed here as a variable. (We assume that in the remainder of this discussion about operators the convention that the symbols 'x' and 'y' are being employed as variables.) The negation operator and the existential quantifier are *one-place operators*, since each of them operates on a single expression. Some operators are *two-place*, *three-place*, and so on. The operator (or connective) standing for disjunction, for example, is a two-place operator. The expressions

on which an operator operates are called *operands* of the operator. Just as a one-place operator is placed to the left or right of its operand, so an n-placed operator is placed to the left or right of the sequence of its n operands. Such a sequence may, for perspicuity, be enclosed in parentheses, and the operands may be separated by commas. A two-place operand is often placed between its two operands, and when this is done some writers enclose the resulting expression in square brackets rather than in parentheses, a procedure that will usually be followed in the present Dictionary. Thus we would write '$[p \& q]$' rather than '$(p \& q)$'. Each operator is limited to use with operands of a specified kind, and if used with another kind the resulting total expression is meaningless or at least not well-formed. In the case of operators containing binding variables, the permitted operands usually are forms associated with functions in the sense of 05.3. For example, the expression '$5x$' is a form associated with the function f that has the property that $f(n) = 5n$ for every positive integer n (assuming that the range of the variable 'x' is taken to be the class of positive integers); and the following operator,

$$\sum_{x=1}^{x=3}$$

may be allowed to operate on '$5x$' to give the expression

$$\sum_{x=1}^{x=3} 5x$$

which denotes the number 30. If an associated form contains just one variable, the form must be operated on by an appropriate operator which contains that same variable if the resulting expression is to be a name or a sentence and not just another form. Similar considerations apply when the form contains more than one variable. For example, the operator

$$\prod_{x,y=1}^{x,y=2}$$

when applied to the form '$[x+y]$' gives a name of the number 72, but if the operator

$$\prod_{y=1}^{y=2}$$

is applied to the same form the result is not a name of a number but rather is a form associated with a function of one argument. In fact the function is the function given by the following equation:

$$f(x) = \prod_{y=1}^{y=2} [x + y].$$

06.2. There is an important operator called the *λ-operator* (*lambda-operator*) which when applied to a form gives a name of the function with which the form is associated (cp. 40.2 ff., 44.1 ff.). If 'x' is used as the binding variable, the operator is written as 'λx'. For example, if the λ-operator 'λx' is applied to the form '$5x$' to give the expression '$\lambda x(5x)$' (inserting parentheses to avoid ambiguity), the result is a name of the function f which is such that $f(n) = 5n$ for every positive integer n (assuming that the range of 'x' is the class of positive integers). Thus we have,

$$(\lambda x(5x))(n) = 5n,$$

and in particular,

$$(\lambda x(5x))(2) = 10.$$

Similarly the expression '$\lambda x(e^x)$' serves as a name of the exponential function, where 'e^x' is an associated form of this function and where the range of 'x' is assumed to be the class of real numbers. In the case of functions of two variables we can use a double λ-operator '$\lambda x \lambda y$'. For example, the expression,

$$\lambda x \lambda y [x + y]$$

is then a name of the addition function. Similarly the expression,

$$\lambda x \lambda y [x + y^2]$$

and the expression,

$$\lambda y \lambda x [x + y^2]$$

are names of two different functions (which are converses of each other). We may illustrate the difference between the two latter functions by saying that $(\lambda x \lambda y [x + y^2])(2, 3)$ is 11, while $(\lambda y \lambda x [x + y^2])(2, 3)$ is 7. There are also triple λ-operators quadruple λ-operators, and so on, for dealing with

functions of three or more variables. The operator '$\hat{x}$' is closely analogous to the λ-operator 'λx' and may, in some cases, be regarded as a special case of the latter (40.7, 51.2). We may read '$\hat{x}$' as "the class of things x which are such that". Thus, '$\hat{x}[x = 8]$' would denote the class of numbers equal to 8, and hence would denote the unit class that has the number 8 as its only member. The operator '$\hat{x}$' is usually applied only to propositional forms (that is, forms associated with propositional functions), while the λ-operator 'λx' can be applied in a more general way.

06.3. Quantifiers are another important kind of operator ([Ch] 41). If 'x' is used as the binding variable, the universal quantifier is usually written as '(x)' or as '$\forall x$'. It may be read as "for all x" or as "for every x". For example, the proposition, "For all x, $x = x$", may be expressed,

$$(x)[x = x].$$

If 'x' is used as the binding variable, the existential quantifier is usually written as '$(\exists x)$' or as '$\exists x$'. It may be read as "for some x" or as "there is an x such that". For example, the proposition, "For some x, $x + 2 = 3$", may be expressed,

$$(\exists x)[x + 2 = 3].$$

For further details regarding quantifiers, see 21 below.

06.4. If 'x' is being used as a binding variable of an operator and if this operator is applied to an expression which contains an occurrence of 'x', this latter occurrence of 'x' is said to be *bound* in (relatively to) the total resulting expression (including the operator). The occurrence of 'x' in the operator itself is also said to be bound in the total resulting expression (and this is true even if there is no occurrence of 'x' in the expression to which the operator is applied). Thus both occurrences of 'x' in '$(\exists x)[x = a]$' are bound in '$(\exists x)[x = a]$'. Furthermore, if an occurrence of a variable is bound in part of some expression, then that occurrence of the variable is also assumed to be bound in the whole expression. For example, since both occurrences of 'x' in '$(\exists x)[x = a]$' are bound in '$(\exists x)[x = a]$', it follows that both occurrences of 'x' in '$[p \supset (\exists x)\{x = a\}]$' are bound in the latter expression. On the other hand, neither of these occurrences of 'x'

Notice that the above statement is referring to the symbol,

=

and *not* to the longer expression,

'='

but it *uses* the latter expression as a *name* for the former. Similarly, the expression written as follows,

'*a*'

is a name of the alphabetically first lower-case italic letter.

08.3. In the present Dictionary there is need also to employ special metalanguage variables in discussing various systems and formalized languages. For this purpose, therefore, every expression consisting of an *italic* letter together with a pair of special quotation marks enclosing it will serve as a metalanguage variable except where there is indication to the contrary in accordance with 08.9 below. An example of a metalanguage variable is the following:

'*a*'.

08.4. Special quotation marks may be placed around not only individual italic letters but also around longer expressions that are built out of these italic letters together with various other symbols (as in 13.25) or English words (as in 10.21). When special quotation marks are used in this way, the total resulting expression, in this Dictionary, is not a name of the enclosed expression but is rather a metalanguage variable referring to (having as values) the expressions obtainable from the enclosed expression by replacing all italic letters by various expressions of an object-language (interpreted system) or of an uninterpreted system. (We usually leave it ambiguous as to which expressions these are, since they might differ from system to system; also, in contexts where no such systems are explicitly indicated, we assume that some system could be made available to provide the requisite expressions.) For example, consider the statement:

If '*p*' and '*q*' are sentences, so is '*p*.*q*'.

This statement means that if any two object-language expressions are sentences (of the object-language), so is the result of placing a period

between them. This usage of special quotation marks was suggested by a somewhat similar usage of Quine ([Q] 33 ff.) which he calls *quasi-quotation*. In his usage quasi-quotation marks are written as follows:

⌜ ⌝

Quine's quasi-quotation marks or corners, moreover, are never placed around individual letters but only around longer expressions.

08.5. When italic letters are used in this Dictionary not enclosed in single quotation marks or in special quotation marks, and not as parts of expressions enclosed in such quotation marks (and not, of course, in or as a word in ordinary discourse), then these italic letters are being used as metalanguage variables that may have as their values various things other than expressions. Their values may be propositions, relations, classes, and so on. For example, consider the statement:

If *p* and *q* are propositions, so is *p*.*q*.

This statement means that the conjunction of any two propositions is itself a proposition. It is not a statement about expressions but about propositions. In any general context of a suitable kind, also, there is to be assumed a certain obvious kind of correspondence between italic letters within special quotation marks and the same italic letters not within quotation marks, as in the following statement:

The expression '$x \in F$' means that *x* is a member of *F*.

This statement would assert, in effect, that if the symbol '$\in$' is placed between two expressions, then the resulting expression means that the thing denoted by the expression to the left of '$\in$' is a member of the thing denoted by the expression to the right of '$\in$'.

08.6. The lower-case Greek letters phi ('ϕ'), psi ('ψ'), and chi ('χ') will be reserved for a special usage in this Dictionary. An expression consisting of any one of these Greek letters followed by an italic letter or a sequence of italic letters will itself be used in very much the same way as a single italic letter but in a slightly more special way. This is perhaps best explained by giving some examples. Consider the following statement:

If 'ϕa' and 'ψb' are sentences, if '*a*' and '*b*' are constants, and if '*x*' is a variable, then '$(x)[\phi x . \psi x]$' is a sentence.

This statement would mean that, given any sentence 'p' within which a constant 'a' occurs one or more times and any sentence 'q' within which a constant 'b' occurs one or more times, then, if 'x' is a variable and if 'r' results from 'p' by replacing 'a' everywhere by 'x' and if 's' results from 'q' by replacing 'b' everywhere by 'x', then '$(x)[r,s]$' is a sentence. (We will also include the case where 'p' is a sentence in which 'a' does not occur at all. In this case 'p' and 'r' would be the same sentence. Similarly we include the case where 'b' does not occur in 'q'. In this case 'q' and 's' would be the same sentence.) Consider also the following statement:

If 'ϕabc' is a sentence, if 'a', 'b', and 'c' are constants and if 'x', 'y', and 'z' are variables, then 'ϕxyz' is a well-formed formula.

This statement would mean that if 'p' is a sentence within which constants 'a', 'b', and 'c' each occurs at least once or possibly not at all, then the result of replacing 'a', 'b', and 'c' everywhere in 'p' respectively by variables 'x', 'y', and 'z' is a well-formed formula.

08.7. Sometimes in this Dictionary certain Greek letters other than phi ('ϕ'), psi ('ψ'), and chi ('χ') will be used in the same way as italic letters. When this is done attention will be called to this fact. (See 52.1.)

08.8. Sometimes triple dots or triple dashes will be used to perform the same service as that performed above by Greek letters. For example, the idea expressed by the statement displayed just above (in 08.6) could equally well be expressed in the following way:

If '$(\ldots a\ldots b\ldots c\ldots)$' is a sentence, if '$a$', '$b$', and '$c$' are constants and '$x$', '$y$', and '$z$' are variables, then '$(\ldots x\ldots y\ldots z\ldots)$' is a well-formed formula.

Also the first statement displayed in 08.6 could be rephrased as follows, using dots and dashes in place of Greek letters:

If '$(\ldots a\ldots)$' and '$(\text{---}b\text{---})$' are sentences, if 'a' and 'b' are constants, and if 'x' is a variable, then '$(x)[(\ldots x\ldots)\cdot(\text{---}x\text{---})]$' is a sentence. (See 21.1.)

08.9. Occasionally the various symbols that have been reserved to serve as metavariables, such as italic letters and certain lower-case Greek letters, must themselves be used not as metavariables but as instances of the notation used by other writers for certain other purposes. In such cases there will be a reference made to this paragraph, and it will be understood that the symbols in question are not being used as metavariables, as, for example, in 25.1 and 61.5.

09. Use of algebraic notation

09.1. Historically, mathematical logic (symbolic logic) had its origin in two rather separate developments. The first of these developments is the old algebra of logic, which began with the work of Boole (19*1*, 19*3*, XVI 224, XXIV 203) and may be considered to have had its culmination in the writings of Schröder (42*7*, 42*10*, 42*12*), Couturat (100*16*), and Huntington (122*1*). The other development began with the work of Frege (49*1*, 49*10*) who was followed by Russell, Hilbert, Carnap, and many other modern logicians. This second development is distinguished by its use of the logistic method (cp. 01.11–01.13). The point of view of this second development is the one in the main adopted in this Dictionary. The logistic method specifies explicitly what statements are treated as meaningful and what inferences are allowable from these statements. Hence it does not leave these two basic considerations to a rather uncertain natural intuition, and it is particularly useful where a system is being specified for the express purpose of proposing or laying down foundations for some discipline, say, mathematics.

09.2. The modern successors of the algebra of logic are themselves of major significance in their own (mathematical) context and retain important applications even to logic as treated by the logistic method. Boolean algebra, for example, which has the calculus of classes as one of its interpretations, has long been an important branch of abstract algebra. There have been proposed recently more comprehensive kinds of "algebraic logic" – in particular, the cylindrical algebras of Tarski (XXII 215 (1, 2)) and the polyadic algebras of Halmos (XXVII 468–470).

09.3. An algebra of propositions, an algebra of classes, and an algebra of (binary) relations were the three branches developed in nineteenth-century algebra of logic. Each of these three branches has its counterpart in modern logic as a certain formalized language. These counterparts of the algebras of propositions, of classes, and of relations will be treated

This statement would mean that, given any sentence 'p' within which a constant 'a' occurs one or more times and any sentence 'q' within which a constant 'b' occurs one or more times, then, if 'x' is a variable and if 'r' results from 'p' by replacing 'a' everywhere by 'x' and if 's' results from 'q' by replacing 'b' everywhere by 'x', then '$(x)[r.s]$' is a sentence. (We will also include the case where 'p' is a sentence in which 'a' does not occur at all. In this case 'p' and 'r' would be the same sentence. Similarly we include the case where 'b' does not occur in 'q'. In this case 'q' and 's' would be the same sentence.) Consider also the following statement:

If 'ϕabc' is a sentence, if 'a', 'b', and 'c' are constants and if 'x', 'y', and 'z' are variables, then 'ϕxyz' is a well-formed formula.

This statement would mean that if 'p' is a sentence within which constants 'a', 'b', and 'c' each occurs at least once or possibly not at all, then the result of replacing 'a', 'b', and 'c' everywhere in 'p' respectively by variables 'x', 'y', and 'z' is a well-formed formula.

08.7. Sometimes in this Dictionary certain Greek letters other than phi ('ϕ'), psi ('ψ'), and chi ('χ') will be used in the same way as italic letters. When this is done attention will be called to this fact. (See 52.1.)

08.8. Sometimes triple dots or triple dashes will be used to perform the same service as that performed above by Greek letters. For example, the idea expressed by the statement displayed just above (in 08.6) could equally well be expressed in the following way:

If '$(\ldots a \ldots b \ldots c \ldots)$' is a sentence, if '$a$', '$b$', and '$c$' are constants and '$x$', '$y$', and '$z$' are variables, then '$(\ldots x \ldots y \ldots z \ldots)$' is a well-formed formula.

Also the first statement displayed in 08.6 could be rephrased as follows, using dots and dashes in place of Greek letters:

If '$(\ldots a \ldots)$' and '$(\text{-}\cdot\text{-}b\text{-}\cdot\text{-})$' are sentences, if '$a$' and '$b$' are constants, and if 'x' is a variable, then '$(x)[(\ldots x \ldots)\cdot(\text{-}\cdot\text{-}x\text{-}\cdot\text{-})]$' is a sentence. (See 21.1.)

08.9. Occasionally the various symbols that have been reserved to serve as metavariables, such as italic letters and certain lower-case Greek letters, must themselves be used not as metavariables but as instances of the notation used by other writers for certain other purposes. In such cases there will be a reference made to this paragraph, and it will be understood that the symbols in question are not being used as metavariables, as, for example, in 25.1 and 61.5.

09. Use of algebraic notation

09.1. Historically, mathematical logic (symbolic logic) had its origin in two rather separate developments. The first of these developments is the old algebra of logic, which began with the work of Boole (19*1*, 19*3*, XVI 224, XXIV 203) and may be considered to have had its culmination in the writings of Schröder (42*7*, 42*10*, 42*12*), Couturat (100*16*), and Huntington (122*1*). The other development began with the work of Frege (49*1*, 49*10*) who was followed by Russell, Hilbert, Carnap, and many other modern logicians. This second development is distinguished by its use of the logistic method (cp. 01.11–01.13). The point of view of this second development is the one in the main adopted in this Dictionary. The logistic method specifies explicitly what statements are treated as meaningful and what inferences are allowable from these statements. Hence it does not leave these two basic considerations to a rather uncertain natural intuition, and it is particularly useful where a system is being specified for the express purpose of proposing or laying down foundations for some discipline, say, mathematics.

09.2. The modern successors of the algebra of logic are themselves of major significance in their own (mathematical) context and retain important applications even to logic as treated by the logistic method. Boolean algebra, for example, which has the calculus of classes as one of its interpretations, has long been an important branch of abstract algebra. There have been proposed recently more comprehensive kinds of "algebraic logic" – in particular, the cylindrical algebras of Tarski (XXII 215 (1, 2)) and the polyadic algebras of Halmos (XXVII 468–470).

09.3. An algebra of propositions, an algebra of classes, and an algebra of (binary) relations were the three branches developed in nineteenth-century algebra of logic. Each of these three branches has its counterpart in modern logic as a certain formalized language. These counterparts of the algebras of propositions, of classes, and of relations will be treated

respectively in Chapters 1, 5, and 6 below. There is also a sense in which first-order functional (or "predicate") logic (treated in Chapter 2 below) may be regarded as a modern descendant of the algebra of relations. (Concerning the relationship between the two, see Tarski VII 38.)

09.4. Both the calculus of classes and the calculus of relations may be viewed as included in set theory, and their notations are generally used in set theory. The operator for abstraction (cp. 06.2), dealt with as a primitive notation in the calculus of λ-conversion and obtainable by (contextual) definition in combinatory logic (Chapter 4 below), provides a means for defining some of the concepts of set theory in terms of functional logic.

09.5. Set theory as an autonomous discipline will not be treated in this Dictionary. But it should be noted that set theory (treated by the logistic method) and type theory (as represented by the functional calculi of higher order, Chapter 3 below) may be regarded as two fairly equally plausible methods of providing a proper foundations of mathematics. For a treatment of type theory from a set-theoretic viewpoint, see Quine II 51. And for general expositions of set theory see Bernays III 49, IX 74, XXII 376; Bernays and Fraenkel XXIV 224; Fraenkel XI 29; *Foundations of Set Theory* by Abraham A. Fraenkel and Yehoshua Bar-Hillel (1958, North-Holland Publ. Co., Amsterdam); and Gödel VI 112.

09.6. In this Dictionary, then, we begin with propositional logic (Chapter 1), followed by functional logic of various orders (Chapters 2 and 3). Chapter 4 deals with combinatory logic and the calculus of λ-conversion. Chapter 5 is concerned with the calculus of classes, and Chapter 6 with the calculus of relations. Chapters 7 and 8 discuss the axiomatization of arithmetic, and Chapter 9 deals with metamathematics.

CHAPTER 1

PROPOSITIONAL CALCULUS

10. Two-valued propositional calculus; an informal explanation

10.1. INTRODUCTION. Two-valued propositional calculus provides a method for specifying some propositions as being logically true and others as being logically false. It can be assumed that there are definite ways of compounding propositions from other propositions. From this standpoint two-valued propositional calculus may be viewed as concerned with certain selected ways of compounding propositions out of other propositions (10.21). The truth or falsity of such a compound proposition may be viewed as depending on the truth or falsity of its component propositions. Furthermore, it is assumed that each proposition is either true or false (but cp. 13 below). There are various varieties of two-valued propositional calculus, but they are all in a fundamental sense equivalent to each other. Some of these use axioms and rules of inference. Others emphasize notation that is useful for expressing the so-called "truth-values" of propositions and for clarifying the dependence of the truth-value of a compound proposition on the truth-values of its component propositions. The approach presented below is of this latter sort. In all this discussion it is important to keep in mind the distinction between the sentence, which is a linguistic or symbolic representation of the proposition, and the proposition itself.

10.2. CONNECTIVES AND TRUTH-FUNCTIONS

10.21. Propositions may be thought of as what (declarative) sentences express (cp. 04). Some propositions, as has been already noted, may be considered to be compounded from other propositions. Sentences expressing compound propositions may be formed from sentences expressing the component propositions by means of appropriate connectives, such as, for example, 'or', 'and', and 'if... then...'. Thus if 'p' and 'q' are

sentences expressing propositions p and q, then 'p or q', 'p and q', and '$if\ p$ then q' are corresponding compound sentences expressing propositions which are related to – we shall say "compounded of" – the propositions p and q in a way which is expressed by the connectives. In symbolic logic it is customary to use special symbols to play the role of these connectives (12.2), and these symbols are then themselves called connectives (06.1). Propositions do not have values in the way functions do, but still we say that every true proposition has the truth-value truth and every false proposition has the truth-value falsehood. It is assumed in two-valued propositional calculus that every proposition dealt with is either true or false. These truth-values may be thought of as attributes of propositions or else as propositions of a special kind which are equivalent to the propositions that "have them" (04.4), or with Frege, as the denotations of sentences (04.3). In any case, the kind of compounding of propositions that is here considered is such that the truth-value of a compound proposition can be determined from a knowledge of the truth-values of the component propositions.

10.22. This latter fact is the starting point of our explanation of the interpretation of the connectives used in propositional logic. These propositional connectives – or perhaps better, sentence connectives – are sometimes taken as having no meaning in isolation (05.1). They may consist of ordinary words or of special symbols. If they consist of ordinary words the words may be conjunctions (such as 'and' and 'or'), or conjunctive phrases (such as 'if... then...'), but may sometimes be verbs ('implies'), verb phrases ('is equivalent to'), or adverbs ('not'). If, using such a connective as 'if... then...', we construct such a form as '$if\ p$ then q', where 'p' and 'q' are variables, then the function of which this form is an associated form (05.4) is said to be the *associated function* of the connective. A sentence connective is understood by understanding its associated function. In some contexts it may be appropriate to regard the connective as a name of the associated function (cp. 32.26); but when this is done the connective is viewed as a functional symbol.

10.23. The associated function of a sentence connective (10.22) is a propositional function which is usually called a *truth-function* and is described by means of a *truth-table* (10.47). Such tables will be shown below

(11) for each of the truth-functions commonly used in propositional logic. These propositional functions are called truth-functions because the truth or falsity of the arguments of such a function determines the truth or falsity of the value of the function.

10.24. Not only are the truth-tables sufficient for defining the associated truth-functions of the connectives of two-valued propositional calculus, but they also provide us with a way of determining the class of theorems of that calculus. Just as a proposition has a truth-value, so in a parallel but slightly different sense of "has", a sentence expressing a proposition has a truth value, in fact the same truth-value as the proposition has which the sentence expresses. Thus the truth-value of a compound sentence can be determined in a way that parallels that for determining the truth-value of a compound proposition. The necessary and sufficient condition for a sentence to be a theorem of two-valued propositional (sentential) calculus is that it be a compound sentence which has the truth-value truth for every possible assignment of truth-values to the simple (non-compound) sentences out of which it is compounded. Such a sentence is often called a *truth-table tautology*. (For an example, see 12.37 below.) In any particular case such sentences would constitute a specific subclass of the well-formed formulas of some formulation of two-valued propositional (sentential) calculus. In saying that a proposition is a theorem of such a calculus, we would ordinarily mean that the proposition is expressed by one of the sentences which is a theorem of the calculus. Such propositions may themselves be called truth-table tautologies.

10.3. Truth-functions of one argument

10.31. We consider first truth-functions of one argument and the connectives with which each of these functions is associated. There are four such truth-functions, since the argument of the function may be true or false and the value of the function may be true or false. (In what follows we use 'p', 'q', and 'r' to refer to propositions.) The two most important of these four functions are:

10.32. *Affirmation.* The affirmation of p states that p is true. That is, the affirmation of p is true if p is true, false if p is false.

10.33 *Negation.* The negation (denial) of p states that p is false. That is, the negation of p is false if p is true, true if p is false.

10.4. Truth-functions of two arguments

10.41. The arguments of a truth-function of two arguments may have four (i.e., 2^2) combinations of truth-values:
1) both arguments true,
2) first argument true, second argument false,
3) first argument false, second argument true,
4) both arguments false.
For each combination of the truth-values of the arguments the value of the function may be either true or false; hence we may have $2^{2^2}=2^4=16$ different truth-functions of two arguments. The most important of these sixteen functions are:

10.42. *Conjunction.* The conjunction of p and q states that p and q are true; that is, the conjunction is true if p and q are both true and is false in any other case.

10.43. *(Non-exclusive) disjunction.* The (non-exclusive) disjunction of p and q states that p or q is true; that is, the disjunction is false if p and q are both false and is true if at least one of them is true.

10.44. *Material implication.* The material implication of q by p states that p materially implies q. To say that p materially implies q amounts to saying that p is false or q is true. In other words, it amounts to saying that it is false that both p is true and q is false. Hence the necessary and sufficient condition for the truth of the proposition asserting that p materially implies q is that at least one of the following two conditions holds: (1) p is false; (2) q is true. Material implication is very useful in logical analysis. Often a sentence, '*If p, then q*', may be regarded as synonymous with 'p materially implies q', but there are cases of the use of 'if... then...' in ordinary language which cannot be adequately analyzed in terms of material implication. As a consequence logicians have sought to define other sorts such as Lewis' *strict implication* (14.1), Burks' *causal implication* (XVI 277), Ackermann's *strenge Implikation* (XXII 327), and Anderson and Belnap's *entailment* (Alan R. Anderson and Nuel D. Belnap Jr., "Tautological Entailments", *Philosophical Studies*, Vol. 13 (1962) pp. 9–24). Related to this need for some other sorts of implication is the fact that a false proposition materially implies every proposition, and the fact that a true proposition is materially implied by every proposition; while in the case of some other kinds of implication we would not wish to assert that a false proposition implies every proposition, nor that a true proposition is implied by every proposition. On the other hand, for the phrase, 'if... then', some such situation can perhaps be allowed, because it seems not unacceptable to say, "If a false proposition is true, then *any* proposition is true", and "If any proposition is true, then a true one is".

10.45. *Inverse material implication.* The inverse material implication of q by p states that p is materially implied by q. That is, the inverse material implication is false if q is true and p false and otherwise is true. Remarks similar to those about material implication (10.44) are relevant also to inverse material implication.

10.46. *Material equivalence.* The material equivalence of p and q states that p and q have the same truth-value; hence the material equivalence is true if p and q are both true or both false and otherwise is false. Often a sentence, 'p if and only if q', may be regarded as synonymous with 'p is materially equivalent to q'. Remarks similar to those about material implication (10.44) are relevant also to material equivalence.

10.47. Truth-functions can also be defined in such a way as to take truth-values as their arguments and values, instead of taking propositions as their arguments and values. (Of course, if truth-values are viewed as a special kind of proposition, as in 04.5, then truth-functions would still take propositions as arguments and values, but only propositions of that special kind.) Truth-tables can be regarded as defining truth-functions of either of these two kinds. In the case of truth-functions which use only truth-values as their arguments and values, the truth-table *explicitly* defines the function by specifying which truth-value *is* the value of the function corresponding to each specification as to which truth-values *are* arguments of the function. In the case of truth-functions which use propositions as their arguments and values, the truth-table *implicitly* defines the function by specifying which truth-value the value of the function *has*

corresponding to each specification as to which truth-values the arguments of the function *have*.

11. Notation for two-value truth-tables

11.1. The truth-tables defining the functions described in 10.3 and 10.4 will be represented symbolically in our explanations as follows:

11.2. We write 't' for truth and 'f' for falsehood. We agree that 't' precedes 'f' in a lexicographic order.

11.3. The truth-table for negation (10.33) will be:

(first form)

p	negation of p
t	f
f	t

(second form)

p	t	f
Negation of p	f	t

and similarly for other truth-functions of one argument.

11.4. The truth-table for conjunction (10.42) will be:

(first form)

p	q	conjunction of p with q
t	t	t
t	f	f
f	t	f
f	f	f

(second form)

p	t	t	f	f
q	t	f	t	f
Conjunction of p with q	t	f	f	f

and similarly for other truth-functions of two arguments.

11.5. We shall give truth-tables for truth-functions in the notation of the "second form" presented above, but abridged by writing only the last line of the table, and adding parentheses, thus:

Affirmation of p	(t f)	(10.32)
Negation of p	(f t)	(10.33)
Conjunction of p with q	(tf ff)	(10.42)
Disjunction of p with q	(tt tf)	(10.43)
Material implication by p of q	(tf tt)	(10.44)
Inverse material implication by p of q	(tt ft)	(10.45)
Material equivalence of p with q	(tf ft)	(10.46)

12. Symbols of propositional calculus

12.1. PROPOSITIONAL VARIABLES. The symbols most frequently used as propositional variables are the letters 'p', 'q', 'r', and 's', and these same letters primed one or more times or with indices. Capitals are used in [HA] (3, 65), [HB] (i 45), [K] (72, 109); Heyting (3852), and Reichenbach (XIV 50, p. 23) use the lower-case letters 'a', 'b', 'c',....

12.2. TABLE OF SYMBOLS

English equivalent	PM	HA HB	Ł	HS	Other notations
not p	$\sim p$	$\bar{p}$	Np	$\bar{p}$	p', $-p$, $\neg p$
p and q	$p.q$ pq	$p\&q$	Kpq	$p \wedge q$	
p or q	$p \vee q$	$p \vee q$ pq	Apq	$p \vee q$	$p+q$
if p then q	$p \supset q$	$p \rightarrow q$	Cpq	$p \rightarrow q$	
p if q	$p \subset q$		Bpq	$p \leftarrow q$	
p if and only if q	$p \equiv q$	$p \sim q$	Epq	$p \leftrightarrow q$	

Under PM we give the notation of *Principia Mathematica* ([PM]), which is used also by [Ch], [R], and numerous other authors (sometimes with slight variations, cp. [K]). This is the notation generally used for explana-

tory purposes in this Dictionary. Under HA we give the notation of [HA] and [HB], which is used also by Gentzen. The notation of Łukasiewicz is given under Ł; this notation makes unnecessary the use of parentheses, dots, etc. (cp. 16). This notation is frequently used by Polish logicians. The notation under HS is used by Hermes-Scholz, and in numerous publications by members of the school of Münster. Notations related to those given under HA and HS are used, e.g., by Tarski, Beth, and also by Kleene, but the latter uses Heyting's negation sign (cp. 15.3).

12.3. COMMENTS ON THE TABLE (12.2)

12.31. We have used the lower-case italic letters 'p' and 'q' as propositional variables throughout the table. The actual letters used as propositional variables differ from author to author.

12.32. There is considerable typographical variety in the symbols '.', '&', '$\vee$', '$\wedge$' and '$\supset$'. For instance, the dot may be heavy or light, round or square, and it may appear on or above the line. The vee or wedge may be light or heavy, small or large, upper or lower case. Where roman capitals are used in 12.2 or 12.5, the authors themselves often use italic capitals.

12.33. The symbol for conjunction is omitted altogether in many standard notations, and the mere juxtaposition of two propositional variables is used in place of an expression consisting of two such variables joined by a conjunction symbol. This convention will generally not be followed in this Dictionary. In [HA] (122) the disjunction symbol, rather than the conjunction symbol, is omitted in this way. In Feys X 100 and Levin XV 69, the symbol for 'neither... nor...' is omitted in this same way.

12.34. '$\sim$' is used for denial or negation in [Ch] (37), [Q] (13), and [R] (12), but for material equivalence in [HA] (4), [HB] (i 47), and [K] (9). In the latter usage a longer symbol, '$\sim\!\!-$', is sometimes preferred.

12.35. '$\rightarrow$' is generally used to express material implication or implication of other kinds, such as Anderson and Belnap's entailment (10.44). Church ([C]77) uses the arrow to express the fact that one expression is an abbreviation for another. See also 96.4.

12.36. '$\equiv$' often expresses material equivalence or equivalence of other kinds. In Curry XVIII 266 it expresses definitional equivalence.

12.37. The following is an example of a truth-table using the connectives '$\sim$' and '$\supset$'.

p	q	$\sim p$	$\sim q$	$p \supset q$	$\sim q \supset \sim p$	$[p \supset q] \supset [\sim q \supset \sim p]$
t	t	f	f	t	t	t
t	f	f	t	f	f	t
f	t	t	f	t	t	t
f	f	t	t	t	t	t

The sentence '$[p \supset q] \supset [\sim q \supset \sim p]$' has the truth-value t for all assignments of truth-values to the sentences 'p' and 'q', and so it is a truth-table tautology (10.24).

12.4. INTERDEFINABILITY. As a consequence of the way certain truth-functions have been defined by truth-tables, certain equivalences hold. For instance, $p \supset q$ is equivalent to $\sim p \vee q$ and to $\sim [p . \sim q]$. And $p . q$ is equivalent to $\sim [\sim p \vee \sim q]$. These equivalences may be used to define implication in terms of negation and disjunction, or in terms of negation and conjunction, and to define conjunction in terms of negation and disjunction. Similarly disjunction can be defined in terms of negation and conjunction.

12.5. CONNECTIVES CORRESPONDING TO THE NEGATIONS OF CERTAIN PROPOSITIONAL FORMS

English equivalent	Church ([Ch] 37)	Other notations Prior XXII 224 (7, 8, 10, 11, 12) [Q] 45	Equivalent notation of this Dictionary
not both p and q	$p \mid q$	Dpq	$\sim [p . q]$
neither p nor q	$p \bar{\vee} q$	Xpq	$\sim [p \vee q]$
		$p \downarrow q$	
p but not q	$p \not\supset q$	Lpq	$\sim [p \supset q]$
q but not p	$p \not\subset q$	Mpq	$\sim [p \subset q]$
p or else q	$p \not\equiv q$	Jpq	$\sim [p \equiv q]$

12.6. MORE-THAN-BINARY CONNECTIVES

12.61. Ternary connectives are seldom used. Church, however, in effect uses a ternary connective when he writes '$[p, q, r]$' to express a proposition equivalent to that expressed by '$[p \supset p].[\sim q \supset r]$' ([Ch] 129).

12.62. The conjunction of p_1 with p_2 with p_3 with ... with p_n is sometimes expressed thus:

$$\prod_{i=1}^{i=n} p_i.$$

12.63. The disjunction of p_1 with p_2 with p_3 with ... with p_n is sometimes expressed thus:

$$\sum_{i=1}^{i=n} p_i.$$

12.7. PROPOSITIONAL CONSTANTS. Use is sometimes made of two propositional constants, which may be translated as *truth* and *falsehood*. The constant translated as *truth* may be thought of as expressing an arbitrary true proposition, and the constant translated as *falsehood* may be thought of as expressing an arbitrary false proposition. On the other hand, if the truth-values truth and falsehood are themselves regarded as special propositions (as suggested in 04.5), then these two constants can be regarded as expressing these two truth-values. In any case, every true proposition is materially equivalent to the proposition expressed by the constant for truth, and every false proposition is materially equivalent to the proposition expressed by the constant for falsehood. Church uses the italic lower-case letters '*t*' and '*f*' as these constants ([Ch] 73). Others write '$\wedge$' for the constant for falsehood and '$\vee$' for the constant for truth (Gentzen 4422, Johansson II 47). Other notations for the constant for falsehood are '*F*' (Curry XVIII 266(2)) and '0' (Wajsberg 4372).

13. Many-valued propositional calculi

13.01. In the classical propositional calculus three conditions are satisfied: (1) all functions are truth-functions, (2) there are exactly two truth-values, and (3) there is exactly one "designated" truth-value, namely

truth. We shall consider non-classical calculi which fail to satisfy one or more of these conditions.

13.02. The *many-valued* calculi satisfy condition (1), but in these calculi there are more than two truth-values. Also there may be more than one designated truth-value, and if so, then the various designated truth-values correspond to different kinds of truth. Modal logics (14) generally fail to satisfy (1).

13.1. TRUTH-VALUES

13.11. Systems with a finite or denumerably infinite number of truth-values have been studied. The truth-values of each such system are completely ordered; that is, for any two truth-values, one is "higher" than the other. These truth-values are usually denoted by numbers, but there is little uniformity among the notations used in the various many-valued calculi.

13.12. For a calculus with a finite number m of truth-values, the most usual notation is as follows: 1 is the highest truth-value, m the lowest. The intermediary truth-values correspond to the natural numbers between 1 and m. Unless otherwise stated, we use this notation here. The rules given can easily be adapted to other notations. (Strictly speaking, the symbols denoting the natural numbers are used, for convenience, to denote truth-values. Sometimes we may speak as if some numbers actually *are* truth-values, but this is not literally true. Notice also that the order of the numbers is opposite to the order of the truth-values.)

13.13. As an alternative notation, for a calculus with a finite number m of truth-values, the highest truth-value may be $m-1$, the lowest one 0, the intermediary truth-values being denoted by the natural numbers between $m-1$ and 0. For a calculus with a finite or denumerably infinite number of truth-values, the highest truth-value might also be expressed by '1', the lowest truth-value by '0', the intermediary truth-values being denoted by rational numbers between 1 and 0 (by all such numbers if the number of truth-values is denumerably infinite).

13.14. Sometimes truth-values may be roughly interpreted as degrees of truth: the highest truth-value as "completely true", the lowest one as

"completely false", the intermediary truth-values as degrees of truth "between" these two extremes. More precise interpretations have been given by Post (2801) and Zawirski (3556).

13.15. The highest truth-value 1 (13.12) is always a designated truth-value, the lowest truth-value m, always undesignated. There is a truth-value s such that truth-values greater than or equal to s are designated, and truth-values less than s are undesignated. Most frequently there is but one designated truth-value, so that $s=1$.

13.16. The number of truth-functions will grow very rapidly with an increasing m. With three truth-values ($m=3$) there are already $3^3=27$ truth-functions of one argument and $3^{3^2}=3^9=19683$ truth-functions of two arguments.

13.17. The preceding considerations about truth-tables (10, 11) hold also for many-valued systems. The method of truth-tables yields, as with two-valued logic, a decision method (10.24). A propositional form is valid if it has a designated truth-value for each of the possible combinations of truth-values of its arguments. We may also transfer to many-valued logic the terminology of sense and denotation. Thus a sentence of many-valued logic could be said to denote a truth-value (cp. 04.2).

13.18. In the following explanations we take as a symbol for a given truth-function an abridged form of the truth-table corresponding to that function (cp. 11.5), supposing the combinations of truth-values to be put in a lexicographic order such that a higher truth-value precedes a lower one, and then writing the values of the function in that order. In this notation, if the truth-table of $F(p)$ is

p	1	2	3
$F(p)$	3	2	1

then the notation for $F(p)$ will be '(3 2 1)'. If the truth-table for $F'(p,q)$ is

p	1	1	1	2	2	2	3	3	3
q	1	2	3	1	2	3	1	2	3
$F'(p,q)$	1	1	1	1	2	2	1	2	3

then the notation for $F'(p,q)$ will be '(111 122 123)'.

13.19. Connectives may be associated with any truth-functions. In actual fact, however, connectives have been associated only with functions analogous to those of classical propositional calculus and some few others.

13.2. Normal connectives of Łukasiewicz

13.21. For any many-valued system with exactly one designated value, Łukasiewicz (1864) has associated connectives with truth-functions such that most of the theorems of two-valued logic are valid in any such system. Since the symbols in these systems are the same as for two-valued logic, we refer back to the table of symbols (12.2) for their various forms. These connectives, described below in 13.22–13.27, will be called normal connectives because of their similarity to connectives of two-valued logic. We mention first the identity function, expressed by the absence of any connective.

13.22. The phrase 'is true' may be considered to express the identity function, since to assert p is equivalent to asserting that p is true. The truth-value of this function is always the same as the truth-value of the argument of the function. For $m=3$ (three-valued logic) the truth-table for the identity function is

(1 2 3).

13.23. If p has n as its truth-value, the value of the negation of p will be $(m+1)-n$, where m is the total number of truth-values. Hence, for $m=3$, the negation function has the truth-table

(3 2 1).

13.24. Let n and n' be respectively the truth-values of p and q in this section and in sections 13.25–13.27 below. Then the *conjunction* of p and q, written '$p.q$', has the lower of the two truth-values n and n'. For $m=3$, the truth-table for conjunction is

(123 223 333).

It can be seen that the conjunction of p and q is never more true than either of the propositions p and q, and also that $p.q$ has a designated truth-value if and only if both p and q have a designated truth-value.

13.25. The (non-exclusive) *disjunction* of p and q, which is written '$p \vee q$', has the higher of the two truth-values n and n' (cp. 13.24). For $m=3$, the truth-table for disjunction is

(111 122 123).

It can be seen that the disjunction of p and q is never less true than either of the propositions p and q, and also that $p \vee q$ has a designated value if and only if at least one of the propositions, p and q, has a designated value.

13.26. For the *implication* of q by p, written '$p \supset q$',

(1) If $n > n'$ (cp. 13.24), then $p \supset q$ has the designated value 1. This is also the case for $p \supset q$ in two-valued logic (cp. 10.44).

(2) If $n < n'$, then $p \supset q$ has the value $1+n'-n$. This is also the case in two-valued logic, for if $n=1$ and $n'=2$, then the truth-value of $p \supset q$ is 2. (Here 2 corresponds to falsity and 1 to truth.)

For $m=3$, the truth-table for implication is

(123 112 111).

13.27. For the *equivalence* of p with q, written '$p \equiv q$',

(1) If $n=n'$ (cp. 13.24), then $p \equiv q$ has the value 1.

(2) If $n \neq n'$, then $p \equiv q$ has the value $1+|n-n'|$.

These are also the case in two-valued logic. For $m=3$, the truth-table for equivalence is

(123 212 321).

It may be stressed that $p \equiv q$ is not equivalent to $[p.q] \vee [\sim p.\sim q]$ in m-valued logic for m greater than 2. In three-valued logic the function which has '$[p.q] \vee [\sim p.\sim q]$' as an associated form has as its truth-table

(123 222 321).

13.28. Despite the analogy between two-valued and many-valued functions, some definitions valid in two-valued logic (12.4) cannot be transferred even to normal connectives (13.21) of many-valued logic. For instance, $p \supset q$, $\sim p \vee q$, and $\sim [p.\sim q]$ are no longer equivalent.

13.29. In one respect two-valued logic is already contained in m-valued logic for m greater than 2, at least if the m-valued logic is "full" in the sense of Słupecki, III 165. For we can define one function in m-valued logic expressing truth and another expressing falsehood. If f_1 is the function for truth, then f_1 is defined in such a way that $f_1(p)$ has a designated value if and only if p itself has a designated value. If f_2 is the function for falsehood, then f_2 is defined in such a way that $f_2(p)$ has a designated value if and only if p does not have a designated value. Since every proposition p has either a designated or a non-designated value in m-valued logic, it is either "true" or "false" in the sense that either $f_1(p)$ or $f_2(p)$ has a designated value, but not both. Viewed in this way, the designated values are seen to represent different subdivisions of the truth notion of two-valued logic, while the undesignated values represent different subdivisions of the falsehood notion of two-valued logic.

13.3. The functions J_k

13.31. Let the following truth-function be associated with $J_k p$ (Rosser and Turquette, XVIII 288, 16 ff.):

$J_k p$ has the value 1 if p has the value k,

$J_k p$ has the value m if p has a value other than k.

Then $J_k p$ may be interpreted as affirming strongly (but not merely affirming as the identity function (13.21) does) that p has the truth-value k. For $m=3$, J_1 is defined by the truth-table

(1 3 3)

i.e., it affirms strongly that p has the value 1 (and denies that it may have any other value). J_2 is defined by the truth-table

(3 1 3)

i.e., it affirms strongly that p has the value 2 (and no other value). Likewise J_3 is defined by the truth-table

(3 3 1)

and it affirms strongly that p has the value 3 (and no other value).

13.32. Thus for $m=3$, the truth-value of $J_1(\sim p)$ is given by the truth-table

(3 3 1).

This affirms strongly that p is (strongly) false and denies that it has any other value.

13.33. Consider $J_1(p,q)$ which has as its truth-values the values given by the truth-table

(133 333 333).

This affirms strongly that p and q are both (strongly) true and denies any other "weaker" situation. There are similar affirmations for disjunction, implication, and equivalence, namely:

$J_1(p \vee q)$ with truth-table: (111 133 133)
$J_1(p \supset q)$ with truth-table: (133 113 111)
$J_1(p \equiv q)$ with truth-table: (133 313 331).

13.4. OTHER CONNECTIVES

13.41. Post (280*1*) and others after him (e.g., Rose XVIII 66) have defined a cyclic negation np which has the value 1 if p has the value m and the value $k+1$ if p has a value k different from m. Thus for $m=3$, the function which has 'np' as an associated form is defined by the truth-table

(2 3 1)

and the function having '$n(np)$' or 'n^2p' as an associated form is defined by the truth-table

(3 1 2).

The function having '$n(n(np))$' or 'n^3p' as an associated form is defined by the truth-table

(1 2 3)

and so on.

13.42. We might also mention the connective 'T' used in axiomatizations of three-valued logic (Słupecki II 46; Rosser and Turquette, XVIII 288, pp. 17, 107). The function having 'Tp' as an associated form is defined by the truth-table

(2 2 2).

Tp has the value 2 whatever the value of p may be. There exist in two-

valued logic similar functions, one always having the value 1 and another always having the value 0, no matter what the truth-value of the argument.

14. Modal logics

14.1. INTRODUCTION. Modal logic is as old as Aristotle. Formalized modal logics were formulated by Lewis as early as 1918 (2159), and many distinct and consistent modal logics have been developed since. Dissatisfaction with the properties of material implication (implication as defined in 10.44) led Lewis to formulate a system in which implication reflected more closely the ordinary non-technical usage of 'if... then...'. His aim was to avoid the so-called "paradoxes" of material implication (10.44), namely, that a false proposition materially implies any proposition, and that a true proposition is materially implied by any proposition. Lewis does in fact avoid these "paradoxes", but analogous "paradoxes" arise on another level, since in his systems a necessary proposition is "strictly implied" by any proposition, and an impossible proposition "strictly implies" any proposition whatever.

14.2. MODAL LOGIC AS AN EXTENSION OF CLASSICAL LOGIC. We confine our present discussion to modal logics which are extensions of classical (two-valued) logic. Fitch has formulated a modal logic which is an extension of intuitionistic logic with quantifiers (XIV 261). (For a revised form of this see 24.4.) We shall not consider here the modal logics of Łukasiewicz (1868) which are many-valued logics with a modal interpretation.

14.3. NECESSITY AND POSSIBILITY IN MODAL LOGICS. We confine ourselves to the discussion of *alethic* modal logics, that is, to logic with modes of truth. (For discussion of the different types of modalities, see Von Wright XVIII 174, p. 2.) Alethic modal logics have the property that they contain not only affirmations such as that some proposition p is true, but contain also stronger affirmations such as that p is necessary and weaker affirmations such as that p is possible. If we assert that p is necessary, then we can assert also that p is true; and if we assert that p is true, then we can assert also that p is possible. 'p is possible' has been symbolized as '$\Diamond p$' (456*1*) or in the Polish notation by 'Mp' (1868). 'p is necessary' is written '$\Box p$' (Barcan XI 96; [F] 64); in the Polish notation it is written 'Lp' or 'Np' or as '$\#p$'.

This affirms strongly that p is (strongly) false and denies that it has any other value.

13.33. Consider $J_1(p.q)$ which has as its truth-values the values given by the truth-table

(133 333 333).

This affirms strongly that p and q are both (strongly) true and denies any other "weaker" situation. There are similar affirmations for disjunction, implication, and equivalence, namely:

$J_1(p \vee q)$ with truth-table: (111 133 133)
$J_1(p \supset q)$ with truth-table: (133 113 111)
$J_1(p \equiv q)$ with truth-table: (133 313 331).

13.4. OTHER CONNECTIVES

13.41. Post (2801) and others after him (e.g., Rose XVIII 66) have defined a cyclic negation np which has the value 1 if p has the value m and the value $k+1$ if p has a value k different from m. Thus for $m=3$, the function which has 'np' as an associated form is defined by the truth-table

(2 3 1)

and the function having '$n(np)$' or 'n^2p' as an associated form is defined by the truth-table

(3 1 2).

The function having '$n(n(np))$' or 'n^3p' as an associated form is defined by the truth-table

(1 2 3)

and so on.

13.42. We might also mention the connective 'T' used in axiomatizations of three-valued logic (Słupecki II 46; Rosser and Turquette, XVIII 288, pp. 17, 107). The function having 'Tp' as an associated form is defined by the truth-table

(2 2 2).

Tp has the value 2 whatever the value of p may be. There exist in two-

valued logic similar functions, one always having the value 1 and another always having the value 0, no matter what the truth-value of the argument.

14. Modal logics

14.1. INTRODUCTION. Modal logic is as old as Aristotle. Formalized modal logics were formulated by Lewis as early as 1918 (2159), and many distinct and consistent modal logics have been developed since. Dissatisfaction with the properties of material implication (implication as defined in 10.44) led Lewis to formulate a system in which implication reflected more closely the ordinary non-technical usage of 'if... then...'. His aim was to avoid the so-called "paradoxes" of material implication (10.44), namely, that a false proposition materially implies any proposition, and that a true proposition is materially implied by any proposition. Lewis does in fact avoid these "paradoxes", but analogous "paradoxes" arise on another level, since in his systems a necessary proposition is "strictly implied" by any proposition, and an impossible proposition "strictly implies" any proposition whatever.

14.2. MODAL LOGIC AS AN EXTENSION OF CLASSICAL LOGIC. We confine our present discussion to modal logics which are extensions of classical (two-valued) logic. Fitch has formulated a modal logic which is an extension of intuitionistic logic with quantifiers (XIV 261). (For a revised form of this see 24.4.) We shall not consider here the modal logics of Łukasiewicz (1868) which are many-valued logics with a modal interpretation.

14.3. NECESSITY AND POSSIBILITY IN MODAL LOGICS. We confine ourselves to the discussion of *alethic* modal logics, that is, to logic with modes of truth. (For discussion of the different types of modalities, see Von Wright XVIII 174, p. 2.) Alethic modal logics have the property that they contain not only affirmations such as that some proposition p is true, but contain also stronger affirmations such as that p is necessary and weaker affirmations such as that p is possible. If we assert that p is necessary, then we can assert also that p is true; and if we assert that p is true, then we can assert also that p is possible. 'p is possible' has been symbolized as '$\Diamond p$' (1956) or in the Polish notation by 'Mp' (1868). 'p is necessary' is written '$\Box p$' (Barcan XI 96; [F] 64); in the Polish notation it is written 'Lp' or 'Np' or as '$\not M\not p$'.

14.4. Modal functions and modalities

14.41. In part following Parry (V 37), we define modal functions in the following manner:

(1) Any propositional variable is an associated form (05.3) of a modal function

(2) If 'P' and 'Q' are associated forms of modal functions, so are '$\Box P$', '$\sim P$', '$P.Q$', '$P \vee Q$', and '$\Diamond P$'.

Then we say that a modality is any modal function whose associated form has no operators other than '$\sim$', '$\Box$', or '$\Diamond$'. Thus '$\Box p$', '$\Diamond p$', '$\Box \sim p$', and '$\Diamond \sim p$' (where 'p' is an object-language variable) all represent modalities and can be translated respectively as 'p is necessary', 'p is possible', 'p is necessarily false', and 'p is possibly false'. Likewise '$\sim \Box p$', '$\sim \Diamond p$', '$\sim \Box \sim p$', and '$\sim \Diamond \sim p$' all represent modalities and can be translated respectively as 'p is not necessary' (equivalent to 'p is possibly false'), 'p is impossible' (equivalent to 'p is necessarily false'), 'p is not necessarily false' (equivalent to 'p is possible'), and 'p is not possibly false' (equivalent to 'p is necessary'). These are all the simple modalities. Other modalities can be represented by '$\Box\Box p$', '$\Box\Diamond p$', '$\Diamond\Diamond p$', '$\Diamond\Box p$', and so on. These multiple modalities will be touched on in 14.53 below.

14.42. When the modal symbols '$\Box$' and '$\Diamond$' are combined not only with '$\sim$' but also with '$.$', '$\vee$', '$\supset$', and '$\equiv$', various modal functions other than modalities are represented. Some of these will now be mentioned.

14.43. '$p \dashv q$' is used as equivalent to '$\sim \Diamond [p.\sim q]$' (Lewis 2159, p. 293) or to '$\Box [p \supset q]$' ([F] 66). The function so represented is called "strict implication". We translate '$p \dashv q$' as 'p strictly implies q' or as 'p necessarily implies q'.

14.44. '$p \equiv q$' ([F] 77) and sometimes '$p = q$' (456*I*) is used as equivalent to '$[p \dashv q].[q \dashv p]$' or to '$\Box [p \equiv q]$'. We translate '$p \equiv q$' as '$p$ is necessarily (or strictly) equivalent to q'. The function so represented is called "strict equivalence".

14.45. '$p \circ q$' (Lewis 2159, p. 293; [F] 75) is sometimes used as equivalent to '$\Diamond [p.q]$'; this expression is to be translated as 'it is possible that both p and q' or as 'p and q are compatible (consistent) with each other'. The function so represented is called "consistency".

14.5. Distinction of systems

14.51. The foregoing might suffice for a rough verbal translation of the most frequently used modal symbols, since most of the known systems of modal logic use the same few types of symbols. It is to be emphasized, however, that infinitely many non-equivalent modal logics can be constructed, and that a given system is susceptible of various interpretations.

14.52. Axioms and rules analogous to those for universally quantified propositions (21) may be given for modal propositions. These rules and axioms may be added to a system of two-valued logic. Some of these axioms and rules are:

(1) '$\Box p \supset p$';

(2) '$[p \dashv q] \supset [\Box p \supset \Box q]$'; and

(3) if 'p' is a theorem, then '$\Box p$' is a theorem.

We may also have the definition of $\Diamond p$ as equivalent to $\sim \Box \sim p$. The analogues for universally quantified propositions are:

(1) '$(x)\phi x \supset \phi x$';

(2) '$(x)[\phi x \supset \psi x] \supset [(x)\phi x \supset (x)\psi x]$'; and

(3) if 'ϕx' is a theorem, then '$(x)\phi x$' is a theorem.

Also, $(\exists x)\phi x$ may be defined as equivalent to $\sim(x)\sim\phi x$. (Compare (1), (1m), (1o), (1p), and (2) of 23.4 with respectively (3a), (3b), (3c), (3d), and (4) of 24.4.)

14.53. Von Wright has constructed three systems of modal logic, M, M', and M'' (XVIII 174), using rules and axioms essentially equivalent to those in 14.52 above. Differences arise between these systems according to the various possibilities of reduction of modalities with two or more occurrences of '$\Box$' or '$\Diamond$' to simpler modalities. In M there are no such rules of reduction. In M' there is an equivalence between '$\Box p$' and '$\Box\Box p$' and between '$\Diamond p$' and '$\Diamond\Diamond p$'. In M'' there is also an equivalence between '$\Box\Diamond p$' and '$\Diamond p$' and between '$\Diamond\Box p$' and '$\Box p$'. Hence in M'' all modalities are reducible to those with zero or one occurrence of the symbol '$\Box$' or '$\Diamond$'. Thus M'' distinguishes only six modalities, whose associated forms are 'p', '$\sim p$', '$\Box p$', '$\Diamond p$', '$\sim \Box p$', and '$\sim \Diamond p$' (where 'p' is an object-language variable). The systems M' and M'' are respectively

equivalent to the systems S4 and S5 of Lewis; the system M is somewhat stronger than the system S2 of Lewis since rule (3) of 14.52 does not hold for S2 (cp. 456*l* appendix).

15. Systems of the intuitionistic type and related systems

15.1. INTRODUCTION. In the systems of the intuitionistic type we are now to consider, just as in the two-valued calculus and in the many-valued calculi, there is no apparatus for the expression of modalities. But the intuitionistic systems differ from the two-valued systems in lacking the principle of excluded middle and from some many-valued systems in lacking the "generalized principle of excluded middle" to the effect that every proposition has one and only one of the truth-values of the system. For these intuitionistic-type systems there are no truth-tables with a finite number of truth-values such that any theorem of the system yields a designated truth-value for any combination of the truth-values of the variables (cp. 418*6*).

15.2. BROUWER-HEYTING INTUITIONISTIC PROPOSITIONAL CALCULUS

15.21. The most important of these systems or calculi is the Brouwer-Heyting propositional calculus (385*2*). This system employs the connective 'not' with one argument and the four connectives 'and', 'or', 'if... then...', and 'if and only if' each with two arguments. The meaning of these connectives can be fully understood only by a study of the way they function in the system. However, the following remarks may serve as an informal elucidation.

15.22. The proposition not-p (that is, the proposition that p is false) is true if and only if p implies falsehood (and a false proposition implies any other proposition). In particular not-p is provable if and only if a contradiction is deducible from p. Therefore, not-not-p is provable whenever a contradiction can be inferred from the assumption of not-p. This situation can frequently arise in the intuitionistic logic without our being able to prove p. Thus for the intuitionist, not-not-p is a weaker proposition than p itself.

15.23. The conjunction of p and q is true if and only if both p and q are true, and it is provable if and only if both p and q are provable.

15.24. The disjunction of p and q is true if and only if at least one of the propositions p and q is true, and it is provable just in case at least one of p and q is provable. In this second respect (as to provability) intuitionistic logic differs sharply from classical (two-valued) logic. For in classical logic we can prove the disjunction of p and q in many cases where we cannot prove either p or q; that is, our resources permit us to establish that one or the other of the two propositions is true, without enabling us to decide which. In particular, we can always prove the disjunction of p with not-p classically, but intuitionistically we can prove this disjunction only if we can decide which of the two alternatives actually holds.

15.25. The implication of q by p is provable if and only if q is deducible from p.

15.26. The equivalence of p and q is provable if and only if p and q are deducible from each other.

15.27. The functions of intuitionistic logic are not interdefinable in the same way as are those of two-valued logic (12.4). For instance, $p \supset q$ is not equivalent to $\neg p \vee q$ or to $\neg[p.\neg q]$. But $\neg p \vee q$ implies $p \supset q$ (in the Brouwer-Heyting logic) and $p \supset q$ implies $\neg[p.\neg q]$, even in minimal logic (15.5). Notation would ordinarily conform with 15.3 below.

15.3. NOTATION FOR THE INTUITIONISTIC CALCULUS

English translation	Heyting (385*2*, p. 43) (385*3*, p. 60)	Kleene ([K] 69, 113)
not-p (it is absurd that p)	$\neg p$	$\neg p$
p and q	$p \wedge q$	$p \& q$
p or q	$p \vee q$	$p \vee q$
if p then q	$p \supset q$	$p \supset q$
p if and only if q	$p \supset\subset q$	$p \sim q$

15.4. INTERPRETATION OF THE INTUITIONISTIC CALCULUS. Gödel has stated a correspondence (or several equivalent correspondences) giving an in-

terpretation of the intuitionistic calculus in the M′ or S4 modal calculus (41812). If in a propositional expression provable in the intuitionistic calculus each expression of any of the forms listed in the upper line of the following table is replaced by an expression of the corresponding form in the second line of the table, the resulting expression will be provable in the M′ or S4 modal calculus (14.53).

$\neg p$	$p \wedge q$	$p \vee q$	$p \supset q$	$p \supset\subset q$
$\sim\Box p$	$\Box p . \Box q$	$\Box p \vee \Box q$	$\Box p \supset \Box q$	$\Box p \equiv \Box q$

We may translate the symbol '$\Box$' by 'it is provable'. Then '$\neg p$', '$p \wedge q$', '$p \vee q$', '$p \supset q$', and '$p \supset\subset q$' will be translated respectively as 'p is not provable', 'p is provable and q is provable', 'p is provable or q is provable', 'if p is provable then q is provable', and 'p is provable if and only if q is provable'.

15.5. MINIMAL AND D-CALCULUS. In the *minimal calculus* of Johansson (II 47), the negation of p may be defined as equivalent to $p \supset \lambda$, where the symbol '$\supset$' denotes the intuitionistic implication and where the symbol 'λ' denotes an arbitrary fixed proposition, which is usually to be interpreted as a proposition considered as absurd in the system. In Curry's D-calculus (XVIII 266) $\neg p$ may be defined as in the minimal calculus, but $p \vee \neg p$ is admitted as an axiom. The notations of the minimal calculus and the D-calculus are the same as those for the intuitionistic calculus.

15.6. THE FITCH SYSTEM

15.61. Among systems lacking excluded middle (15.1) another example is the system of Fitch's book [F]. This system is similar in some ways to the intuitionistic system. One difference is that the intuitionists equate '$\neg p$' with 'p implies a falsehood', while Fitch distinguishes the two. Suppose for example that we are investigating a certain property P of numbers. Suppose we succeed in showing that the assumption that every number has the property P leads to a contradiction, but yet we cannot exhibit a number which lacks the property P. Then both the intuitionist and Fitch (if they were satisfied with the methods used in the derivation of the contradiction) could infer that q implies a falsehood, where q is the proposition that every number has the property P. The intuitionist could also infer the proposition not-q, but Fitch could infer this only if a particular number has been exhibited which lacks the property P. Thus a proposition not-p can be established for the intuitionist by assuming p and deriving a contradiction. But for Fitch this is not so

15.62. *Notation for Fitch's propositional calculus*

English translation	Fitch ([F] 53, 32, 43, 14, 37)
not-p (it is false that p)	$\sim p$
p and q	$p \& q$
p or q	$p \vee q$
if p then q	$p \supset q$
p if and only if q	$p \equiv q$

16. Scope of logical operators

16.1. INTRODUCTION. Symbolic logic needs notations to indicate unambiguously the scope of the logical operators, i.e., to indicate the extent of the expressions which are the operands of a given operation. The need of such a notation arises with propositional logic, but we give here the principles for such notation for other parts of logic as well.

16.11. Our principal concern, with respect to the propositional calculus, is with logical operations such as negation and conjunction which are expressed by means of connectives (operators not involving binding variables (06.1)). But in later parts we shall consider also operators that contain binding variables.

16.12. An operation is expressed by means of an expression called an operator. If there is but one operand, the operator is usually *prefixed* to the argument: modern symbolic logic seldom uses operators as suffixes, such as the '2' and '3' in '$(a^2)^3$' (but cp. 12.2 and 66.19, for example). A (two-place) operator with two operands may be prefixed to its operands (for example, see the Polish notation of 16.6), but it is more usually written as an "infix" between the operands, as, for example, the '$\vee$' in '$p \vee q$'.

16.13. Some operations which are explicitly designated in some systems are indicated in others by simple juxtaposition. Such is the case with the conjunction symbol between propositions (see 12.33).

16.2. Brackets

16.21. Brackets are used to indicate the extent of the operands, much as is done in algebra. These brackets are omitted when there is no danger of ambiguity, e.g., with an expression consisting of one letter. Thus we write '$p \vee q$' and not '$[p] \vee [q]$'.

16.22. There are different forms of brackets such as round brackets or parentheses, square brackets, braces, and brackets of the form: $\langle\ \rangle$. The use of these different forms of brackets with different kinds of operations does not concern us here. A problem which does concern us here is that of indicating the scope of brackets when they occur in such a manner that there are brackets within brackets.

16.23. This distinction of scope is in general not indicated by the form or size of the brackets used, as is sometimes done in elementary mathematical handbooks. In general, a single form and size of brackets is used. For example, in many formulations of propositional logic, parentheses are the only type of bracket used. Where only a single type of bracket is used, the delimitation of operands is automatically indicated by the way in which brackets are paired or included in one another's scopes, so that for each left-hand bracket we find a corresponding right-hand bracket which is its unique mate.

16.3. Seniority among operations

16.31. In ordinary algebra '$(-ab=c)$' is interpreted as equivalent to '$((-(ab))=c)$'. Thus the operation indicated by '$=$' is there considered as more important than or senior to the operation indicated by '$-$', and this latter is senior to the operation (multiplication) indicated by juxtaposition. It is customary in some kinds of systems to arrange the operators (and the operations they represent) in order (or "rank") of seniority. If there is ambiguity as to the scope of some operator because of lack of brackets indicating such scope, and if the ambiguity can be resolved by giving a "senior" operator a larger scope and a "junior" operator a smaller scope,

then this resolution of the ambiguity is the preferable one. If there are several such "preferable" resolutions of the ambiguity, then we must resort to further principles, discussed below (16.51) for resolving the remaining ambiguity. For example, in the expression '$p \equiv q \vee r \vee s$' we might try out the following ways of grouping and indicating scopes:
(1) $[p \equiv q] \vee [r \vee s]$
(2) $[p \equiv [q \vee r]] \vee s$
(3) $p \equiv [q \vee [r \vee s]]$
(4) $p \equiv [[q \vee r] \vee s]$.
If '$\equiv$' is regarded as senior to '$\vee$', then methods (3) and (4) are preferable to methods (1) and (2) because both (3) and (4) give the occurrence of the operator '$\equiv$' a larger scope than either of the occurrences of the operator '$\vee$'. In fact, for this reason, any grouping other than (3) and (4) would be wrong. In order to choose between (3) and (4), however, we must use conventions that assert a preference for grouping to the left or grouping to the right. If grouping to the left is the preferred (or tacitly understood) grouping, then (4) is the correct resolution of the original ambiguity.

16.32. Rules of seniority are in use in almost all systems, except in those employing the Polish notation (16.6) or in systems of combinatory logic (Chapter 4). The most common rules of seniority (analogous to those for algebra) are the following:
(a) an operation indicated by juxtaposition is junior to any other;
(b) an operation indicated by a prefixed connective or operator is senior to an operation indicated by juxtaposition;
(c) an operation indicated by an infix connective or operator is senior to one indicated by juxtaposition or by a prefixed connective.

16.33. Other rules of seniority are frequently involved. The introduction of each work should be consulted for the statement of its rules of seniority.

16.4. Punctuation

16.41. To avoid excessively many brackets and even to avoid brackets altogether, use is frequently made of a punctuation replacing brackets, and this is done under two forms.

16.42. In Peano (71*14*), in [PM] 9, and in works inspired by [PM], the

signs of punctuation consist of one or several heavy dots. A group of (one or more) dots is attached to an operator to indicate the scope of that operator. A larger group of dots will be said to be a "stronger" punctuation than a smaller group of dots. If two groups of the same number of dots are respectively attached to two operators of different seniority, the group attached to the senior operator will be said to be the stronger punctuation. Dots may be used to indicate scope as follows:

(a) Dots may be placed before and after an infix connective or operator such as '$\vee$' or '$\supset$'. Then the scope of the connective or operator extends backwards (to the left) from the left-hand group of dots until the beginning of the expression or a left-hand bracket or an equally strong or stronger punctuation than that group of dots is reached. Similarly the scope of the connective or operator extends forward (to the right) from the right-hand group of dots until the end of the expression or a right-hand bracket or an equally strong or stronger punctuation than that group of dots is reached. For example, '$r \supset [p \supset q . \supset p : \supset . s \supset t]$' would be interpreted as '$r \supset [[[p \supset q] \supset p] \supset [s \supset t]]$'.

(b) Dots may be placed after a prefixed operator such as '$(\exists x)$' (cp. 21.1). Then the scope of the operator extends forward as in (a). For example, '$(x) . \phi x \supset \psi x . \supset \chi x$' would be interpreted as '$(x)[\phi x \supset \psi x] \supset \chi x$'.

(c) If an operator is indicated by juxtaposition there is only one group of dots, placed between the operands, and the scope extends backwards and forwards as in (a). Also in case of an operator which is itself a single dot, this dot may be replaced by a single group of dots to indicate the scope of the operator. For example, '$p : q \supset . p \supset q$' would be interpreted as '$p . [q \supset [p \supset q]]$'.

16.43. The rule given above is the rule for [PM] and for works depending on it. In recent American works (cp. [Ch] 75) single dots rather then multiple dots are used for punctuation. The convention is that if an operator is followed by a dot, the scope extends (to the right) until the end of the expression or a right-hand bracket is reached. To indicate the scope backwards (to the left) we must rely on brackets or on the rule of association to the left (16.5). This rule is not found in [PM]. Refinements of these procedures have been introduced by various authors. The introduction of each work should be consulted for the exact statement of the rule followed

16.5. Rules of abbreviation

16.51. Many American authors now admit a general rule of *association to the left* (cp. [Ch] 74). This may be explained as follows. If the rules of seniority and the rules for dots do not suffice to indicate scopes of operators in an expression because of omitted brackets, then it is understood that brackets are to be restored by inserting them to the left as far as possible compatibly with the nature of the operators concerned. Hence '$p \vee q \vee r$' would be interpreted as '$[p \vee q] \vee r$' and '$p \equiv q \equiv r \supset s$' would be interpreted as '$[[p \equiv q] \equiv r] \supset s$'.

16.52. [PM] has no such rule of association to the left, but uses (especially in the logic of relations (e.g., [PM]*34.22)) various rules "for the avoidance of brackets" which amount in fact to special cases of association to the right.

16.53. Older works, among them [PM], also use abbreviations having their analogues in mathematics, such as that of '$p \equiv q \equiv r$' for the conjunction of '$p \equiv q$' and '$q \equiv r$' ([PM] *4.02) and of '$p \supset q \supset r$' for the conjunction of '$p \supset q$' and '$q \supset r$' ([PM] *3.02).

16.6. Polish parenthesis-free notation. Łukasiewicz has devised a notation which does away with the use of parentheses (1866). Each connective or operator has a fixed number of arguments, and each such connective and operator is written as an immediate prefix of its operands. Instead of '$\supset$', there is a connective 'C' which takes two arguments. Thus instead of '$p \supset q$', the notation 'Cpq' would be used. Similarly, '$p \vee q$' is written as 'Apq', and '$p . q$' as 'Kpq'. For example, the expression,

$$[p . q] \supset [p \vee q]$$

would be written as

$$CKpqApq$$

and the expression,

$$[p \supset [q \supset r]] \supset [[p \supset q] \supset [p \supset r]]$$

would be written as,

$$CCpCqrCCpqCpr.$$

signs of punctuation consist of one or several heavy dots. A group of (one or more) dots is attached to an operator to indicate the scope of that operator. A larger group of dots will be said to be a "stronger" punctuation than a smaller group of dots. If two groups of the same number of dots are respectively attached to two operators of different seniority, the group attached to the senior operator will be said to be the stronger punctuation. Dots may be used to indicate scope as follows:

(a) Dots may be placed before and after an infix connective or operator such as '$\vee$' or '$\supset$'. Then the scope of the connective or operator extends backwards (to the left) from the left-hand group of dots until the beginning of the expression or a left-hand bracket or an equally strong or stronger punctuation than that group of dots is reached. Similarly the scope of the connective or operator extends forward (to the right) from the right-hand group of dots until the end of the expression or a right-hand bracket or an equally strong or stronger punctuation than that group of dots is reached. For example, '$r \supset [p \supset q . \supset p : \supset . s \supset t]$' would be interpreted as '$r \supset [\{[p \supset q] \supset p\} \supset [s \supset t]]$'.

(b) Dots may be placed after a prefixed operator such as '$(\exists x)$' (cp. 21.1). Then the scope of the operator extends forward as in (a). For example, '$(x) . \phi x \supset \psi x . \supset \chi x$' would be interpreted as '$(x)[\phi x \supset \psi x] \supset \chi x$'.

(c) If an operator is indicated by juxtaposition there is only one group of dots, placed between the operands, and the scope extends backwards and forwards as in (a). Also in case of an operator which is itself a single dot, this dot may be replaced by a single group of dots to indicate the scope of the operator. For example, '$p : q \supset . p \supset q$' would be interpreted as '$p . [q \supset [p \supset q]]$'.

16.43. The rule given above is the rule for [PM] and for works depending on it. In recent American works (cp. [Ch] 75) single dots rather then multiple dots are used for punctuation. The convention is that if an operator is followed by a dot, the scope extends (to the right) until the end of the expression or a right-hand bracket is reached. To indicate the scope backwards (to the left) we must rely on brackets or on the rule of association to the left (16.5). This rule is not found in [PM]. Refinements of these procedures have been introduced by various authors. The introduction of each work should be consulted for the exact statement of the rule followed.

16.5. Rules of abbreviation

16.51. Many American authors now admit a general rule of *association to the left* (cp. [Ch] 74). This may be explained as follows. If the rules of seniority and the rules for dots do not suffice to indicate scopes of operators in an expression because of omitted brackets, then it is understood that brackets are to be restored by inserting them to the left as far as possible compatibly with the nature of the operators concerned. Hence '$p \vee q \vee r$' would be interpreted as '$[p \vee q] \vee r$' and '$p \equiv q \equiv r \supset s$' would be interpreted as '$[[p \equiv q] \equiv r] \supset s$'.

16.52. [PM] has no such rule of association to the left, but uses (especially in the logic of relations (e.g., [PM]*34.22)) various rules "for the avoidance of brackets" which amount in fact to special cases of association to the right.

16.53. Older works, among them [PM], also use abbreviations having their analogues in mathematics, such as that of '$p \equiv q \equiv r$' for the conjunction of '$p \equiv q$' and '$q \equiv r$' ([PM] *4.02) and of '$p \supset q \supset r$' for the conjunction of '$p \supset q$' and '$q \supset r$' ([PM] *3.02).

16.6. Polish parenthesis-free notation. Łukasiewicz has devised a notation which does away with the use of parentheses (1866). Each connective or operator has a fixed number of arguments, and each such connective and operator is written as an immediate prefix of its operands. Instead of '$\supset$', there is a connective 'C' which takes two arguments. Thus instead of '$p \supset q$', the notation 'Cpq' would be used. Similarly, '$p \vee q$' is written as 'Apq', and '$p . q$' as 'Kpq'. For example, the expression,

$$[p . q] \supset [p \vee q]$$

would be written as

$$CKpqApq$$

and the expression,

$$[p \supset [q \supset r]] \supset [[p \supset q] \supset [p \supset r]]$$

would be written as,

$$CCpCqrCCpqCpr.$$

CHAPTER 2

THE FIRST-ORDER FUNCTIONAL CALCULUS

20. Propositional functions of individuals

20.1. In the propositional calculus we were concerned with a kind of propositional functions called truth-functions (10.23). The propositional calculus was formulated in Chapter 1 by use of truth-tables. This calculus could also have been formulated by use of axioms and rules of inference (01.12). The latter method or extensions of it, as in natural-deduction techniques ([F] 20), are ordinarily used in the case of the *first-order functional calculus* and *functional calculi of higher order* (Chapter 3). These calculi deal with, in addition to truth-functions, functions of certain other kinds not amenable to truth-table analysis. In the first-order functional calculus these other functions are *propositional functions of individuals*, also to be called simply *functions of individuals*. Functions of this kind presuppose a category of so-called *individuals* (cp. 05.7). Whether these "individuals" are really individuals in some specific philosophical meaning of the term is not especially important. Almost any class could be chosen to serve as the class of "individuals" considered here, and the choice can even be left unspecified. The domain of definition of every one-argument function of individuals is this class of individuals, and the domain of definition of every n-argument function of individuals, where $n>1$, is the class of ordered n-tuples of individuals. The values of these functions of individuals, on the other hand, are propositions. Hence we may view these functions as attributes or properties (or classes) of individuals and as relationships (or relations) among individuals (05.7–05.9). Just as the propositional calculus is concerned primarily with propositions, so the first-order functional calculus is concerned primarily with propositions and with these propositional functions of individuals. Some of the propositions in the first-order functional calculus are values of these functions of individuals and hence are analyzable into the result of applying such functions to individuals, whereas in the propositional calculus any such analysis of propositions is ignored. Sometimes the term 'predicate calculus' is used instead of the term 'functional calculus'. This is often done by writers who use 'predicate' as synonymous with 'propositional function' (for example, [HB] and [K]) or as synonymous with 'symbol for propositional function' (for example, Carnap). Quine uses the term 'predicate' in a still different way (XII 17).

20.2. Special variables are used to range over the class of individuals. Usually lower-case italic letters, with or without numerical subscripts, are used for this purpose. Those are often taken from the end of the alphabet ([Ch] 169; [Q] 69), but Kleene ([K] 143) takes them from the beginning of the alphabet, and Hilbert-Bernays ([HB] i 96) use letters from the beginning and the middle of the alphabet for free variables (cp. 06.4, 21.6) and those from the end of the alphabet for bound individual variables. Special individual constants (names of individuals) are also sometimes used.

20.3. Special variables ranging over functions of individuals are usually used as free variables, but never as bound variables, in the first-order functional calculus; and constants serving as names of functions of individuals are sometimes used. Function symbols of at least one of these two kinds must be used, and both kinds may be used. The prefixing of such a function symbol to an individual variable or constant (or to n such argument symbols in the case of n-argument functions) gives rise to an *atomic formula* of the calculus. Propositional variables and propositional constants may also be used and are the only other kind of atomic formulas. (Propositions may be viewed as being zero-argument functions of individuals, so that propositional variables and constants may be viewed as functional variables and constants of a special kind.) If 'F' is a name of a one-argument function F of individuals, and hence a name of an attribute (property) of individuals, and if 'a' is a name of an individual a, then '$F(a)$' expresses the proposition that a has the attribute F. The terms 'attribute' and 'property' are being used here more or less interchangeably. If 'G' is a name of a two-argument function G of individuals (and hence a two-termed relationship between individuals), and if 'a' and 'b' are respective names of individuals a and b, then '$G(a, b)$' expresses the proposition that a bears the relationship G to b.

20.4. Some writers call functional variables "predicate variables" (cp. 20.1). [HA] (65) speaks of predicate variables '$F(.)$', '$G(.,.)$', '$H(.,..)$', and so on. The symbolism using the dot is employed in the explanations to show somehow that the 'F', 'G', and 'H' represent predicate variables when the 'F', 'G', or 'H' is accompanied by the proper number of variables as indicated by the number of dots in the parentheses following the given letter. A similar explanatory use of formulas with empty places (*Leerstellen*) may be found in Carnap (3523, p. 3) and may be compared with the use of the expressions '$\phi\hat{x}$' (*20) and '$\phi(\hat{x}, \hat{y})$' (*21) in [PM].

20.5. Church ([Ch] 169) indicates the number of arguments a function symbol takes by superscript numbers. Thus he uses 'F^1', 'G^1', 'F_1^1', ..., as functional variables for one-argument functions, 'F^2', 'F_1^2', 'G_2^2', ..., as functional variables for two-argument functions, and so on. The superscripts are usually omitted if there is no danger of ambiguity. Instead of symbols with superscripts, different series of variables might be employed for singulary, binary ..., functional variables.

21. Quantifiers

21.1. In the following discussion we use *formula* for what is often called a *well-formed formula*. Starting with atomic formulas, more complex formulas are built up by use of some or all of the operators (connectives) used in the propositional calculus, together with two kinds of operators called *quantifiers*: universal quantifiers and existential quantifiers. If 'x' is a variable whose range is the class of individuals, a universal quantifier is usually written as '(x)' ([PM]) or as '$\forall x$' (Gentzen 4422) and is read as 'for every individual x' or as 'for all individuals x'. An existential quantifier is usually written as '$(\exists x)$' ([PM]), or as '$\exists x$' (Gentzen 4422), or as '(Ex)' ([HA], [HB], [R]) and is read as 'for some individual x' or as 'there is an individual x such that'. Here the words 'some' and 'an' mean the same as the phrase 'at least one'. Thus if 'F' and 'G' are names of one-argument functions of individuals, and if 'x' is an individual variable, then '$(x)[F(x) \supset G(x)]$' would mean that for every individual x, if x has the attribute F, then x has the attribute G, or that every individual that has the attribute F has the attribute G. The effect of applying a quantifier '(x)' to a formula '$(\ldots x \ldots)$' (cp. 08.8) containing a free occurrence of 'x' (but in which no other variable occurs free) is to construct a sentence '$(x)(\ldots x \ldots)$' which in effect asserts that the function having '$(\ldots x \ldots)$' as an associated form is universal (is an attribute of all individuals). Thus a universal quantifier serves as a device for asserting, in effect, that an attribute of individuals (a propositional function of individuals) is universal. In a similar way an existential quantifier serves as a device for asserting, in effect, that an attribute of individuals (a propositional function of individuals) is non-empty.

21.2. On the supposition that the universe has a finite number n of individuals, '$(x)F(x)$' can be viewed as roughly equivalent to '$F(x_1).F(x_2).F(x_3).\ldots.F(x_n)$'. Similarly '$(\exists x)F(x)$' can be viewed as roughly equivalent to '$F(x_1) \vee F(x_2) \vee F(x_3) \vee \ldots \vee F(x_n)$'. (We employ triple dots here to indicate omission of terms of a sequence according to the usual convention; but this usage is not to be confused with the usages described in 08.8 or in 20.4.) To provide for the case of an infinity of individuals, the functional calculus of the first order has to be derived from axioms giving to formulas with quantifiers properties such that, in the special case of a finite number of individuals, the equivalences just noted would obtain. Hence, quantifiers may be considered as operators which express a kind of generalized operation of (infinite) conjunction ("logical multiplication") and disjunction ("logical addition"). This is reflected in the notations '$\prod_x$' (Schröder 427, 4210, 4212) or '$\prod x$' (Łukasiewicz) or '$\bigcap_x$' (Tarski 2859) for the universal quantifier, and '$\sum_x$', '$\sum x$', and '$\bigcup_x$' for the existential quantifier by the same authors respectively.

21.3. The application of a quantifier to any formula generates another formula. Hence, formulas may have multiple quantifiers, e.g., '$(x)(y)$', '$(\exists x)(\exists y)$', '$(x)(\exists y)$', or '$(\exists x)(y)$'. [PM] *11 uses the abbreviation '(x, y)' for '$(x)(y)$', '$(\exists x, y)$' for '$(\exists x)(\exists y)$', and similarly with more than two quantifiers.

21.4. The following infix operators are definable by means of the quantifiers:

'$\phi x \supset_x \psi x$' for '$(x)[\phi x \supset \psi x]$' ([PM] *10.02)
'$\phi x \equiv_x \psi x$' for '$(x)[\phi x \equiv \psi x]$' ([PM *10 03)

and similarly with two or more quantifiers. E.g.,

$$\text{`}\phi x \supset_{x,y} \psi xy\text{'} \quad \text{for} \quad \text{`}(x, y)[\phi xy \supset \psi xy]\text{'} \quad \text{([PM]*11.05)}$$
$$\text{`}\phi xy \equiv_{x,y} \psi xy\text{'} \quad \text{for} \quad \text{`}(x, y)[\phi xy \equiv \psi xy]\text{'} \quad \text{([PM]*11.06)}.$$

The operators `$\supset_x$', `$\supset_{x,y}$', ..., are said to represent *formal implication* and `$\equiv_x$' and `$=_{x,y}$', ..., *formal equivalences*. Church ([Ch] 171) gives the following abbreviation also:

$$\text{`}\phi x \mid_x \psi x\text{'} \quad \text{for} \quad \text{`}(x)[\phi x \mid \psi x]\text{'}.$$

21.5. It is usually agreed that if `$B$' is a formula and `x' is an individual variable, then `$(x)B$' and `$(\exists x)B$' are formulas, even if `$x$' does not occur in `B'. Some writers, however, require `$x$' to occur free in `B'.

21.6. In conformity with 06.4 and [Ch] 170, an occurrence of a variable `$x$' in a formula `A' is called a *bound occurrence* of `$x$' with respect to (and in) `A' if it is an occurrence in a well-formed part of `$A$' (here a part of `A' which is a formula) of the form `$(x)B$' or `$(\exists x)B$'; otherwise it is called a *free occurrence* of `$x$' with respect to (and in) `A'. The *bound variables* of `$A$' are those which have bound occurrences in `A', and the *free variables* of `$A$' are those which have free occurrences in `A'.

21.7. The closure ([Q] 79) of a formula in the first-order calculus is obtained by prefixing to the formula, for each variable free in that formula, a universal quantifier containing that variable. The quantifiers are prefixed in a row, sometimes according to a rule that determines their order. When Quine encloses a formula schema in his "corners" or "quasi-quotes", `⌜' and `⌝' ([Q] 35), and prefixes the sign of assertion, `⊢', to this whole expression, he is asserting that every formula which is an instance of the schema is a theorem and also that the closure of every such formula is a theorem ([Q] 88)

22. Tables of symbols

TABLE A

Author (pages)	Individual variables (20.2)	Functional variables (20 3)	Atomic formulas (20 3)
[Ch] 169, 169, 169	$x, y, z, \ldots$	$F^1, G^2, \ldots$	$F^1(x), G^2(x, y), \ldots$
[HA] 58, 65, 65	$x, y, z, \ldots$	$F(.), G(.,.), \ldots$	$F(x), G(x, y), \ldots$
[HB] i 96, i 89, i 89	$x, y, z, \ldots$ (bound) $a, b, c, \ldots$ (free)	$A, B(\,)$, $C(\,,\,), \ldots$	$A(x), B(x, y), \ldots$
[K] 143	$a, b, c, \ldots$	$A, B,$ $A(\,), \ldots$	$A, B, A(a),$ $B(c, d), \ldots$
[Q] 69, 120	$x, y, \ldots$		$x \in y$
[R] 87, 208	$x, y, \ldots$ $x_1, x_2, \ldots$		$x \in y$
Other notations	$x, y, \ldots$	$a, b, c, \ldots$ $r, s, t, \ldots$	ax rxy

TABLE B

Author (pages)	Universal quantifier	Existential quantifier
[Ch] 171, 171	(x)	$(\exists x)$
[HA] 58, 59	(x)	(Ex)
[HB] i 95, i 95	(x)	(Ex)
[K] 69, 70	$\forall x$	$\exists x$
[Q] 69, 102	(x)	$(\exists x)$
[R] 88, 90	(x)	(Ex)
Other notations	$\wedge_x$	$\vee_x$
(cp. 21.2)	$\wedge x$	$\vee x$
	$\prod_x$	$\sum_x$
	$\prod x$	$\sum x$
	$\bigcap_x$	$\bigcup_x$

23. An example of a first-order functional calculus

23.1. We now present an example of a first-order functional calculus to illustrate the methods that are sometimes used for defining such concepts as *well-formed formula*, *axiom*, and *theorem*, relatively to such a calculus.

23.2. Let 'x_1', 'x_2',..., serve as variables for individuals, and 'P_1', 'P_2',..., as propositional variables. Also let 'F_1^1', 'F_2^1',..., serve as one-argument functional variables, 'F_1^2', 'F_2^2',..., serve as two-argument functional variables, and so on. We define the class of well-formed formulas (wffs) of the calculus as follows:

(1) Every propositional variable is an atomic wff.

(2) If 'u' is a variable for individuals and if 'f' is a one-argument functional variable, then '$f(u)$' is an atomic wff; also if 'u' and 'v' are variables for individuals and 'g' is a two-argument functional variable, then '$g(u, v)$' is an atomic wff; and similarly for three-argument functional variables, four-argument functional variables, and so on.

(3) Every atomic wff is a wff.

(4) If 'A' and 'B' are wffs, so are '$\sim A$', '$[A.B]$', '$[A \vee B]$', and '$[A \supset B]$'.

(5) If 'u' is a variable for individuals and if 'A' is a wff, then '$(u)A$' and '$(\exists u)A$' are wffs.

(6) 'A' is an atomic wff only if its being so follows from (1) or (2) above; and it is a wff only if its being so follows from (1)–(5) above.

23.3. For example, 'P_2' and '$F_1^3(x_1, x_2, x_1)$' are wffs, and indeed atomic wffs, and '$(\exists x_2)P_2$', '$(x_1)F_1^3(x_1, x_2, x_1)$', '$[(\exists x_2)P_2 \supset (x_1)F_1^3(x_1, x_2, x_1)]$', and '$\sim(x_2)(x_1)F_1^3(x_1, x_2, x_1)$' are wffs. Outermost brackets will hereafter be omitted.

23.4. We will employ infinitely many axioms. The class of axioms is a class of wffs defined as follows:

(1) If 'A', 'B', and 'C' are wffs, and if 'u' is a variable for individuals, then the following wffs are axioms:

(1a) '$A \supset [B \supset A]$';

(1b) '$[A \supset [B \supset C]] \supset [[A \supset B] \supset [A \supset C]]$';

(1c) '$[A.B] \supset A$';

(1d) '$[A.B] \supset B$';

(1e) '$A \supset [B \supset [A.B]]$';

(1f) '$A \supset [A \vee B]$';

(1g) '$B \supset [A \vee B]$';

(1h) '$[A \supset C] \supset [[B \supset C] \supset [[A \vee B] \supset C]]$';

(1i) '$[A \supset \sim A] \supset \sim A$';

(1j) '$\sim A \supset [A \supset B]$';

(1k) '$A \vee \sim A$';

(1l) '$(u)A \supset B$', where for some variable 'v' for individuals the following conditions are satisfied: No free occurrence of 'v' in 'A' is part of a wff of the form '$(u)C$' or '$(\exists u)C$' in 'A', and the result of replacing all free occurrences of 'u' in 'A' by occurrences of 'v' is the wff 'B';

(1m) '$(u)[A \supset B] \supset [(u)A \supset (u)B]$';

(1n) '$A \supset (u)A$', where 'u' is not free in 'A';

(1o) '$B \supset (\exists u)A$', where the conditions in (1l) are satisfied;

(1p) '$(u)[A \supset B] \supset [(\exists u)A \supset (\exists u)B]$';

(1q) '$(\exists u)A \supset A$', where 'u' is not free in 'A'.

(2) If 'A' is an axiom and if 'u' is a variable for individuals, then '$(u)A$' is an axiom.

(3) 'A' is an axiom only if its being so follows from (1) and (2).

23.5. The class of theorems is a class of wffs defined as follows:

(1) Every axiom is a theorem.

(2) If 'A' and 'B' are wffs such that 'A' and '$A \supset B$' are theorems, then 'B' is a theorem.

(3) 'A' is a theorem only if its being so follows from (1) and (2).

23.6. Examples of axioms:

$$P_2 \supset [P_3 \supset P_2],$$
$$F_1^1(x_1) \supset [F_2^1(x_3) \supset F_1^1(x_1)].$$
$$(x_1)[F_1^1(x_1) \supset [F_2^1(x_3) \supset F_1^1(x_1)]],$$
$$(x_2)F_1^2(x_2, x_3) \supset F_1^2(x_1, x_3),$$
$$(x_3)[(x_2)F_1^2(x_2, x_3) \supset F_1^2(x_1, x_3)],$$
$$[P_1.F_1^1(x_1)] \supset P_1,$$
$$(x_1)[[P_1.F_1^1(x_1)] \supset P_1],$$
$$[[P_1.F_1^1(x_1)] \supset P_1] \supset [P_3 \supset [[P_1.F_1^1(x_1)] \supset P_1]],$$

$$(x_1)\{[[P_1 . F_1^1(x_1)] \supset P_1] \supset [P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1]]] .$$
$$(x_1)\{[[P_1 . F_1^1(x_1)] \supset P_1] \supset [P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1]]] \supset$$
$$[(x_1)[[P_1 . F_1^1(x_1)] \supset P_1] \supset (x_1)[P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1]]] .$$

23.7. Examples of theorems that follow from the above axioms:

$$P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1] .$$
$$(x_1)[[P_1 . F_1^1(x_1)] \supset P_1] \supset (x_1)[P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1]] .$$
$$(x_1)\{P_2 \supset [[P_1 . F_1^1(x_1)] \supset P_1]] .$$

23.8. It can be shown that if 'A' is a theorem and 'u' is a variable for individuals, then '$(u)A$' is a theorem (for example, see Fitch XIV 261). As an instance of this, let 'A' be the first example in 23.7 and let '$(u)A$' be the third example.

24. Quantifiers in non-classical logic

24.1. Functional calculi – and in particular first-order functional calculi – may be constructed which are extensions of the various non-classical propositional calculi. Some of these functional calculi have been extensively investigated. The same quantifiers are used as in classical functional logic, but the meaning of these quantifiers is never precisely the same as in classical logic. This becomes clear if, even without inspecting systems of axioms in detail, we consider the interpretation of universal quantification as infinite conjunction and consider existential quantification as infinite disjunction (21.2). In many-valued logic, conjunctions and disjunctions are to be interpreted as has been previously explained (13.2).

24.2. It has been pointed out that a disjunction is provable in intuitionistic logic only if at least one of its disjuncts is provable. In fact, in order to prove the disjunction, it is necessary to know *which* of its disjuncts is provable. Similarly, in intuitionistic logic, an existential sentence, '$(\exists x)(\ldots x\ldots)$' is provable only if some sentence of the form '$(\ldots a\ldots)$' is provable for a particular 'a'; and in order to prove an existential sentence, it is necessary to be able to specify this 'a'. This similarity between disjunction and existential sentences is connected with the fact that existential quantification may be viewed as infinite disjunction (21.2).

24.3. An example of an *intuitionistic* first-order functional calculus is obtainable by dropping axioms of the form '$A \vee \sim A$' (under (1k) of 23.4) from the example of a first-order functional calculus described in section 23. In the resulting system it is not possible to prove every wff of the form '$A \vee \sim A$', so the system may be said to lack the principle of excluded middle. However, if 'A' is provable, so is '$A \vee \sim A$', and if '$\sim A$' is provable so is '$A \vee \sim A$'. This system is equivalent to the system described by Heyting in 3852 and 3853.

24.4. An example of a *modal* first-order functional calculus is obtainable by extending the example given in section 23 in the following way:

(1) In defining the class of wffs we add the stipulation that if 'A' is a wff, so are '$\Box A$' and '$\Diamond A$'.
(2) Modify (6) of 23.2 appropriately.
(3) In defining the class of axioms we add the stipulation that if 'A' and 'B' are wffs, and if 'u' is a variable for individuals, then the following wffs are axioms:

(3a) '$\Box A \supset A$',
(3b) '$\Box [A \supset B] \supset [\Box A \supset \Box B]$',
(3c) '$A \supset \Diamond A$',
(3d) '$\Box [A \supset B] \supset [\Diamond A \supset \Diamond B]$',
(3e) '$\Box \sim A \supset \sim \Diamond A$',
(3f) '$\sim \Diamond A \supset \Box \sim A$',
(3g) '$(u)\Box A \supset \Box (u)A$',
(3h) '$\Diamond (\exists u)A \supset (\exists u)\Diamond A$'.

(4) In defining the class of axioms, we also add the stipulation that if 'A' is an axiom, so is '$\Box A$'.
(5) Modify (3) of 23.4 appropriately.

24.5. In the above modal first-order functional calculus it can be shown that if 'A' is a theorem, so is '$\Box A$'. If all the axioms for quantifiers are omitted, the system becomes equivalent to Von Wright's system M (cp. 14.53). If the axioms for excluded middle (in (1k) of 23.4) are omitted, the system becomes an intuitionistic modal first-order functional calculus. The axiom (3h) is sometimes referred to as "the Barcan formula" (XI 96).

25. First-order functional calculus with identity

25.1. The first-order functional calculus can be extended by adding to the function symbols a special constant for the two-argument function *identity*. We would then include among the wffs such formulas as '$I(x, y)$' ([Ch] 282) and '$x=y$'. (See 08.9 with respect to 'I'.)

25.2. The intuitive properties of identity are embodied in the two principles:
(a) Every individual is identical with itself.
(b) If two individuals are identical (and hence not really "two"), then every property that either has, the other has.

25.3. In order to extend the first-order functional calculus so as to include identity as a function having the above properties, we add to the set of axioms all formulas of the form '$u=u$', where 'u' is a variable for individuals, and all formulas of the form '$[u=v]\supset[A\supset B]$', where 'u' and 'v' are variables for individuals and where the formula 'B' is obtainable from the formula 'A' by replacing one or more occurrences of 'u' that are free in 'A' by occurrences of 'v' in such a way that the latter occurrences of 'v' are free in 'B'. An example of such a formula is:

$$[x_1 = x_2] \supset \{[x_1 = x_1] \supset [x_2 = x_1]\}.$$

26. Definite descriptions

26.1. Roughly speaking, a *definite description* is a phrase or a symbolic equivalent of a phrase beginning with the definite article 'the', provided that the phrase may be construed as being of the form '(the (unique) x such that ...)'. (Here the triple dots stand for some statement about x.)

26.2. '(The (unique) x such that)' is rendered by the operator '$(\imath x)$' in [PM] (*14.01), by 'ι_x' in [HB] (i 383), and by 'ιx' in [R] (182).

26.3. Any affirmation containing the phrase '(the x such that ϕx)' seems to presuppose that there is a unique x such that ϕx, that is, that there is an x such that ϕx and there is but one such x. This presupposition of

unique existence is expressed by '$\mathrm{E!}(\imath x)\phi x$' in [PM] (*14.02) and by '$\exists!x\phi x$' in [K] (199). More specifically, in [PM] *14.02 the expression '$\mathrm{E!}(\imath x)\phi x$' is treated as an abbreviation for '$(\exists y)(x)[\phi x\equiv[x=y]]$'. (See also 26.7.)

26.4. Different authors follow different conventions regarding the exact meaning to be attributed to '$(\imath x)\phi x$'.

26.41. According to [HB] (i 384) '$(\imath x)\phi x$' may be treated as a name of an individual if and only if there is an x and only one x such that ϕx.

26.42. According to Frege, [Ch] (41), and others, two cases are to be considered. If there is exactly one x such that ϕx, then '$(\imath x)\phi x$' denotes that x. If not, '$(\imath x)\phi x$' has some artificial denotation, e.g., the null class.

26.43. [PM] (*14) gives a contextual definition of '$(\imath x)\phi x$' by defining the result of placing '$(\imath x)\phi x$' in a context. According to this definition, '$(\ldots(\imath x)\phi x\ldots)$' is defined as '$(\exists y)\{(x)[\phi x\equiv[x=y]].(\ldots y\ldots)\}$'. In other words, to say, "So-and-so is true of the thing that has such-and-such a property" is to say "One and only one thing has such-and-such a property, and so-and-so is true of that thing".

26.5. There may be some ambiguity as to the extent of the context being considered, because if '$(\imath x)\phi x$' appears within a sufficiently large context, it appears also, at the same time, within smaller contexts which are part of the larger context. This ambiguity may be resolved by a convention that requires the context to be chosen as small as possible, or one that requires the context to be chosen as large as possible, or one that indicates explicitly the extent of the context. For example, consider '$\sim G((\imath x)F(x))$'. If the context of the definite description is chosen as small as possible, then '$\sim G((\imath x)F(x))$' is defined as '$\sim(\exists y)\{(x)[F(x)\equiv[x=y]].G(y)\}$'. But if the context of the definite description is chosen as large as possible, then '$\sim G((\imath x)F(x))$' is defined as '$(\exists y)\{(x)[F(x)\equiv[x=y]].\sim G(y)\}$'. A special kind of operator '$[(\imath x)\phi x]$' is sometimes used to indicate the extent of the context of a definite description and is placed at the beginning of the context. For example, '$\sim G((\imath x)F(x))$' is written as '$[(\imath x)F(x)]\sim G((\imath x)F(x))$'

if the context of the definite description is being taken as large as possible, while '$\sim G((\imath x)F(x))$' is written as '$\sim[(\imath x)F(x)]G((\imath x)F(x))$' if the context is being taken as small as possible. The intended extent of the context is often called the *scope* of the definite description '$(\imath x)\phi x$'. A more economical notation is to write '$^{\imath x}\sim G((\imath x)F(x))$' instead of '$[(\imath x)F(x)]\sim G((\imath x)F(x))$', and to write '$\sim{}^{\imath x}G((\imath x)F(x))$' instead of '$\sim[(\imath x)F(x)]\ G((\imath x)F(x))$', assuming that different variables are associated with different definite descriptions. In the system of [PM] the extent of the scope makes no difference if '$\mathrm{E}!(\imath x)\phi x$' can be proved. In such a case the definite description is in effect acting like a name of the thing described. In quantified modal logic a more complicated criterion than '$\mathrm{E}!(\imath x)\phi x$' is required for this purpose.

26.6. In [PM] when the scope is not indicated it is to be chosen as small as possible (compatible with not replacing defined symbols by their defining symbols). Thus '$(\imath x)\phi x\neq(\imath x)\phi x$' abbreviates '$(\exists y)[(x)[\phi x\equiv[x=y]].[y\neq y]]$', while '$\sim[(\imath x)\phi x=(\imath x)\phi x]$' abbreviates '$\sim(\exists y)[(x)[\phi x\equiv[x=y]].[y=y]]$'.

26.7. Although '$\mathrm{E}!a$' in [PM] is meaningless if 'a' is not a definite description, this limitation could be avoided by treating '$\mathrm{E}!a$' as an abbreviation for '$[a=a]$'. Then '$\mathrm{E}!(\imath x)\phi x$' would be an abbreviation for '$[(\imath x)\phi x=(\imath x)\phi x]$', and this, in turn, by 26.43 above, would be an abbreviation for '$(\exists y)[(x)[\phi x\equiv[x=y]].[y=y]]$', which in the system of [PM] is logically equivalent to '$(\exists y)(x)[\phi x\equiv[x=y]]$', thereby agreeing with the [PM] definition of '$\mathrm{E}!(\imath x)\phi x$' stated above in 26.3. Thus the latter definition becomes superfluous and the treatment of definite description becomes more uniform.

27. Quasi-definite descriptions

27.1. Just as '$(\imath x)\phi x$' may be taken to mean the same as 'the x such that ϕx', so also '$\eta_x\phi x$' is a formal notation used by [HB] (ii 10, 11) to mean the same as 'an x such that ϕx'. It is assumed that if '$(\exists x)\phi x$' is a provable formula, then '$\eta_x\phi x$' is a constant and '$\phi(\eta_x\phi x)$' is a provable formula. '$\eta_x\phi x$' may be called a *quasi-definite description*, just as '$(\imath x)\phi x$' is called a definite description. So, for example, if '$(\exists x)[F(x).H(x)]$' is a provable formula, then it is assumed that the formula '$F(\eta_x[F(x).H(x)]).H(\eta_x[F(x).H(x)])$' is provable too.

27.2. The expression '$\varepsilon_x\phi x$' is defined as an abbreviation for '$\eta_x[(\exists y)\phi y\supset\phi x]$'. Then since '$(\exists x)[(\exists y)\phi y\supset\phi x]$' is provable, so is '$(\exists y)\phi y\supset\phi(\varepsilon_x\phi x)$', and '$\varepsilon_x\phi x$' is a constant. But '$\phi(\varepsilon_x\phi x)\supset(\exists y)\phi y$' is also provable (cp. 23.4 (1o)), so that '$(\exists y)\phi y\equiv\phi(\varepsilon_x\phi x)$' is provable. On the other hand, '$(\exists y)\phi y\equiv\phi(\eta_x\phi x)$' is not provable in general.

27.3. Since '$\varepsilon_x\phi x$' is treated as a constant when 'ϕx' contains no other free variables than 'x', it is clear that the use of this notation in the way described involves supposing that when there are many values of 'x' for which 'ϕx' is true, the choice of one unique one can always be made (uniformly). And indeed, if we allow 'x' to be not merely an individual variable as in [HB] but a variable of any type (in the sense of the next chapter), then '$(\exists y)F(y)\supset F(\varepsilon_xF(x))$', where '$y$' is a variable of the same type as 'x', and 'F' is a variable of appropriate type, is one way of stating what is called the *axiom* (or axiom schema) *of choice*. (Compare axioms 11^{ε} of Church V 114.)

27.4. The formula '$(y)\phi y\equiv\phi(\varepsilon_x\sim\phi x)$' can be shown to be provable by the same general method for proving '$(\exists y)\phi y\equiv\phi(\varepsilon_x\phi x)$'.

if the context of the definite description is being taken as large as possible, while '$\sim G((\imath x)F(x))$' is written as '$\sim[(\imath x)F(x)]G((\imath x)F(x))$' if the context is being taken as small as possible. The intended extent of the context is often called the *scope* of the definite description '$(\imath x)\phi x$'. A more economical notation is to write '$^{x}\sim G((\imath x)F(x))$' instead of '$[(\imath x)F(x)]\sim G((\imath x)F(x))$', and to write '$\sim^{x}G((\imath x)F(x))$' instead of '$\sim[(\imath x)F(x)]\,G((\imath x)F(x))$', assuming that different variables are associated with different definite descriptions. In the system of [PM] the extent of the scope makes no difference if '$\mathrm{E}!(\imath x)\phi x$' can be proved. In such a case the definite description is in effect acting like a name of the thing described. In quantified modal logic a more complicated criterion than '$\mathrm{E}!(\imath x)\phi x$' is required for this purpose.

26.6. In [PM] when the scope is not indicated it is to be chosen as small as possible (compatible with not replacing defined symbols by their defining symbols). Thus '$(\imath x)\phi x\neq(\imath x)\phi x$' abbreviates '$(\exists y)[(x)[\phi x\equiv[x=y]].[y\neq y]]$', while '$\sim[(\imath x)\phi x=(\imath x)\phi x]$' abbreviates '$\sim(\exists y)[(x)[\phi x\equiv[x=y]].[y=y]]$'.

26.7. Although '$\mathrm{E}!a$' in [PM] is meaningless if 'a' is not a definite description, this limitation could be avoided by treating '$\mathrm{E}!a$' as an abbreviation for '$[a=a]$'. Then '$\mathrm{E}!(\imath x)\phi x$' would be an abbreviation for '$[(\imath x)\phi x=(\imath x)\phi x]$', and this, in turn, by 26.43 above, would be an abbreviation for '$(\exists y)[(x)[\phi x\equiv[x=y]].[y=y]]$', which in the system of [PM] is logically equivalent to '$(\exists y)(x)[\phi x\equiv[x=y]]$', thereby agreeing with the [PM] definition of '$\mathrm{E}!(\imath x)\phi x$' stated above in 26.3. Thus the latter definition becomes superfluous and the treatment of definite description becomes more uniform.

27. Quasi-definite descriptions

27.1. Just as '$(\imath x)\phi x$' may be taken to mean the same as 'the x such that ϕx', so also '$\eta_x\phi x$' is a formal notation used by [HB] (ii 10, 11) to mean the same as 'an x such that ϕx'. It is assumed that if '$(\exists x)\phi x$' is a provable formula, then '$\eta_x\phi x$' is a constant and '$\phi(\eta_x\phi x)$' is a provable formula. '$\eta_x\phi x$' may be called a *quasi-definite description*, just as '$(\imath x)\phi x$' is called a definite description. So, for example, if '$(\exists x)[F(x).H(x)]$' is a provable formula, then it is assumed that the formula '$F(\eta_x[F(x).H(x)]).H(\eta_x[F(x).H(x)])$' is provable too.

27.2. The expression '$\varepsilon_x\phi x$' is defined as an abbreviation for '$\eta_x[(\exists y)\phi y\supset\phi x]$'. Then since '$(\exists x)[(\exists y)\phi y\supset\phi x]$' is provable, so is '$(\exists y)\phi y\supset\phi(\varepsilon_x\phi x)$', and '$\varepsilon_x\phi x$' is a constant. But '$\phi(\varepsilon_x\phi x)\supset(\exists y)\phi y$' is also provable (cp. 23.4 (1o)), so that '$(\exists y)\phi y\equiv\phi(\varepsilon_x\phi x)$' is provable. On the other hand, '$(\exists y)\phi y\equiv\phi(\eta_x\phi x)$' is not provable in general.

27.3. Since '$\varepsilon_x\phi x$' is treated as a constant when 'ϕx' contains no other free variables than 'x', it is clear that the use of this notation in the way described involves supposing that when there are many values of 'x' for which 'ϕx' is true, the choice of one unique one can always be made (uniformly). And indeed, if we allow 'x' to be not merely an individual variable as in [HB] but a variable of any type (in the sense of the next chapter), then '$(\exists y)F(y)\supset F(\varepsilon_xF(x))$', where '$y$' is a variable of the same type as 'x', and 'F' is a variable of appropriate type, is one way of stating what is called the *axiom* (or axiom schema) *of choice*. (Compare axioms 11[e] of Church V 114.)

27.4. The formula '$(y)\phi y\equiv\phi(\varepsilon_x\sim\phi x)$' can be shown to be provable by the same general method for proving '$(\exists y)\phi y\equiv\phi(\varepsilon_x\phi x)$'.

CHAPTER 3

FUNCTIONAL CALCULI OF HIGHER ORDER. THE THEORY OF TYPES

30. Orders of calculi

30.1. There are many logical statements of quite an elementary sort which cannot be adequately formalized within the first-order functional calculus (unless variables for individuals are allowed to range over properties as well as individuals (cp. 20.2)). Such, for example, is the statement that any two properties (of individuals) have an intersection; that is, that for any two properties F and G there is a property H such that $H(x)$ is true for just those things x for which $F(x)$ and $G(x)$ are true. Using the notations '(F)' and '$(\exists F)$' for 'for all properties F' and 'there is a property F' respectively, where 'F', 'G', and 'H' are variables for properties of individuals, we arrive at the following formulation:

$$'(F)(G)(\exists H)(x)[H(x) \equiv [F(x) . G(x)]]' .$$

30.2. The calculus which is just like the first-order functional calculus except that the variables for propositional functions of individuals can be bound by quantifiers is called the *second-order functional calculus*. Just as the following formula is provable in the first-order functional calculus:

$$'(x)F(x) \supset F(y)'$$

('Whatever is true of all individuals is true of any individual y'), so every formula of the following form is provable in the second-order functional calculus:

$$'(F)F(x) \supset G(x)'$$

('If an individual has every property then the individual has any property G'). Similarly for functions of more than one argument. However, if 'P' is a variable, the formula

$$'(F)P(F) \supset P(G)'$$

('Whatever is true of all functions of individuals is true of any function of individuals G') is not provable in second-order functional calculus, since that calculus does not have available any such free variable 'P' for one-argument propositional functions of propositional functions of individuals, that is, for properties of properties of individuals.

30.3. We are thus led to construct the *third-order functional calculus*, containing all the free and bound variables of the calculus of second-order, and in addition, free variables for functions of functions of individuals. These additional function variables might be of many kinds, depending on the number of arguments of the functions they range over and the kinds of arguments of those functions (e.g., whether a given argument is itself a function of one argument, a function of two arguments, and so on. (For further details see Church X 19, pp. 104–105.)

30.4. In the fourth-order calculus, quantifiers are allowed which involve the newly introduced functional variables of the third-order calculus. And so on for each higher order of functional calculus. Each calculus of an odd order, $2n+1$, introduces functional variables of a new kind, and each calculus of an even order, $2n+2$, introduces quantifiers involving the variables first introduced in the calculus of order $2n+1$.

30.5. The functional calculus F^{ω} of order ω is obtained by combining all the functional calculi of finite order.

30.6. Within a second-order functional calculus F^2, there is, as a subcalculus, a "singulary" second-order functional calculus $\mathrm{F}^{2,1}$, and also a "binary" second-order functional calculus $\mathrm{F}^{2,2}$, and so on, such that all functions in the singulary functional calculus are functions of one argument, all functions in the binary functional calculus are functions of two arguments, and so on. Within an n-th order functional calculus F^n, there is a singulary n-th order calculus $\mathrm{F}^{n,1}$, a binary n-th order functional calculus $\mathrm{F}^{n,2}$, and so on. In particular, there is a singulary calculus $\mathrm{F}^{\omega,1}$, of order ω, combining the calculi $\mathrm{F}^{1,1}$, $\mathrm{F}^{2,1}$, $\mathrm{F}^{3,1}$, and so on.

30.7. The functional calculi of first and higher order may involve constants as well as variables. For each kind of variable there may be a

corresponding kind of constant. Just as we have functions of one argument, of two arguments, and so on, we can also regard propositions as functions of zero arguments, so that variables ranging over zero-argument functions may then serve as propositional variables. These zero-argument functions may for convenience be thought of as functions of individuals.

30.8. It is possible in F^2 to define identity of individuals in the Leibnizian fashion: thus $'x=y'$ may be defined as $'(F)[F(x) \supset F(y)]'$, somewhat as is done in [PM] *13.01 (cp. 25.1). And similarly for identity of functions of individuals in F^4, and so on.

31. The theory of types

31.1. The distinction among orders of functional calculi is intimately connected with the distinction among *types* of functional variables and constants. We are here concerned with the notational problems raised by the theory of types, and we need not dwell upon its origin or upon its dispensability or indispensability.

31.2. According to the theory of types, every variable and constant belongs to some fixed category called a *type*. A formula is not well-formed, according to this theory, unless each of the expressions in it that denotes a function is of the appropriate type, as determined by the types of the expressions that denote arguments of that function and the types of expression that denote values of it. By a "functional expression" or a "functional symbol" we will mean a symbol or expression which denotes a function (or which would denote a function if all free variables are replaced in it by suitable constants). The type of a functional expression depends on (and is always different from) the types of its argument symbols. In the *simple theory of types* this type depends *solely* on the types of the argument symbols and the type of the value symbols (symbols for the values of the function). Thus, in the case of a function of n arguments, if the respective types of the n argument symbols have been completely specified, as well as the type of the value symbols, then the type of the functional expression itself has been completely specified (cp. 32). In the *ramified* (or *branched*) *theory of types*, the specification of type depends on the nature of the functional expression itself as well as upon

the types of the argument symbols and in some cases also on the types of the value symbols (cp. 31.3). The functional calculus based on the simple theory of types is, in effect, the same as the functional calculus F^ω of order ω. The notion of type, in a slightly different sense, can be applied to functions themselves. Thus, if a functional expression is of some type, we say that the function denoted by that functional expression is also (in a different sense) of that type.

31.3. The *ramified* or *branched theory of types* differs from the simple theory of types in important respects ([PM] Chapter II of Introduction to First Edition). In formulating the ramified theory of types it is convenient to assign types not only to variables and constants, but also to various complex functional expressions. We here consider only the case where the functional expressions denote propositional functions. The concept of the "order" of a type is introduced. This is a further way of distinguishing types, beyond the distinction made in the simple theory of types. The various orders are expressed numerically as 0, 1, 2, 3, ... The type of a functional expression (and in particular the *order* of that type) is made to depend not only on the types of its argument symbols (and its order must be greater than the orders of types of such argument symbols), but also on the types of variables and constants appearing in the functional expression. The order of the type of a functional expression is the least order greater than the order of the type of any variable occurring bound in it, and not less than the order of the type of any variable occurring free in it or of any constant occurring in it; cp. Fitch IV 97. Thus, in the branched theory of types, in the case of a propositional function of n arguments, if the respective types of the n argument symbols have been specified and if the order of the functional expression has been specified, then the type of the function expression has, in effect, been specified. To make this clearer, consider the following example: Napoleon had two properties, among others: the property of being an emperor and the property of having all the properties required for being a great general. Both of these are properties of individuals. Hence, they are expressed by predicates of the same type in simple theory of types. But the ramified theory distinguishes between the orders of the types of these two predicates because one of them involves a quantifier containing a variable of a type of the same order as the type of the other predicate. Thus in the

terminology of the ramified theory of types, the two properties would be, more specifically, the first-order property of being an emperor and the second-order property of having all the first-order properties required for being a great general. (By an n-th order property is meant a property whose type has order n.) Though the types of these two properties are types of properties of individuals, still they are different types, since one is a first-order type and the other is a second-order type.

31.4. There have also been totally type-free theories proposed, as, for example, in [F]. Others ([Q] 155, [R] 205) make use of "stratification" rules which do not attribute one fixed type to each variable and constant.

32. Explicit indication of type

32.1. It is often desirable to indicate explicitly the type of a variable or constant. Various notations are used for this purpose.

32.2. The Church notation for the simple theory of types (cp. Church V 114) consists of subscripts added to each variable or constant; there may also be subscripts accompanying connectives (32.26). The subscripts may serve as names for types in explanations.

32.21. The type of a sentence or of a propositional variable is indicated by omicron, 'o'; the type of a constant denoting an individual or of a variable for individuals is indicated by iota, 'ι'.

32.22. If 'a' and 'b' indicate types, then '(ab)' indicates the type of a (one-argument) functional expression whose argument symbols have the type indicated by 'b' and whose values are expressed by symbols of the type indicated by 'a'. (We may write 'ab' for '(ab)' when no ambiguity results from doing so.) Hence, $o\iota$ is the type of a variable for a propositional function of individuals, and $o(o\iota)$ is the type of a variable for a propositional function of propositional functions of individuals.

32.23. This same method may be further elaborated to deal with functions of more than one argument. If 'a', 'b', and 'c' indicate types, then '(abc)' indicates the type of a two-argument function whose respective argument symbols have the types indicated by 'b' and 'c', and whose values are expressed by symbols of the type indicated by 'a'. (We may write 'abc' for '(abc)' when no ambiguity results from doing so.) For example, a two-argument propositional function of individuals is said to be of type $o\iota\iota$. A two-argument propositional function of two-argument propositional functions of individuals would be of type $o(o\iota\iota)(o\iota\iota)$. If '$a$', '$b$', '$c$', and '$d$' indicate types, then '$(abcd)$' indicates the type of a three-argument function whose respective argument symbols have the types indicated by 'b', 'c', and 'd', and whose values are expressed by symbols of the type indicated by 'a'. For example, a three-argument propositional function of individuals would be of type $o\iota\iota\iota$. A similar procedure can be used for functions of four or more arguments. Observe that a specification of the type of a function entails (and, conversely, is entailed by) a specification of the types of the respective arguments of the function and of the values of the function. It is a rule of the simple theory of types that atomic formulas be of the forms 'p_a', '$x_{ab}(y_a)$', '$x_{abb}(y_a, z_b)$', and so on, where 'p_a', 'y_a', 'x_{ab}', 'x_{abb}', 'z_b' are constants or variables of the types indicated by the subscripts.

32.24. The term 'type' is ambiguous in the respect that it may be applied to a symbol for a function or individual or to the function or individual itself, but these two meanings of 'type' are parallel and distinct, and in practice this ambiguity causes little difficulty.

32.25. In connection with the use of the λ-operator (06.2, 40.2 ff.) it should be noted that if an associated form '$(\ldots x\ldots)$' of some function is of type a and if the variable 'x' in the form is of type b, then '$\lambda x(\ldots x\ldots)$', which denotes the function and which is the result of applying the operator 'λx' to '$(\ldots x\ldots)$', is of type ab. Hence the function denoted by '$\lambda x(\ldots x\ldots)$' is of type ab, using the term 'type', now, as applying to the function itself rather than to the expression denoting the function (cp. 32.24). Similarly if '$(\ldots x\ldots y\ldots)$' is of type a and if 'x' and 'y' are variables of respective types 'b' and 'c', then '$\lambda x\lambda y(\ldots x\ldots y\ldots)$' can be regarded as being of type abc. Special attention should be paid to the situation arising when the double λ-operator is defined by means of two uses of the single (ordinary) λ-operator, so that '$\lambda x\lambda y(\ldots x\ldots y\ldots)$' is defined as '$\lambda x(\lambda y(\ldots x\ldots y\ldots))$', as is customary in combinatory logic (41.3, 44.5). In this

case, if 'x', and 'y', and '$(\ldots x\ldots y\ldots)$' are of respective types b, c, and a, then '$\lambda y(\ldots x\ldots y\ldots)$' is of type ac, and hence '$\lambda x(\lambda y(\ldots x\ldots y\ldots))$' is of type $(ac)b$, so that '$\lambda x\lambda y(\ldots x\ldots y\ldots)$' would be of type $(ac)b$, instead of abc. But this disparity causes no difficulty in combinatory logic because the type abc can there be defined as being the type $(ac)b$, and we could even treat the type-indicating notation 'abc' as an abbreviation for the type-indicating notation '$(ac)b$'. On the other hand, Church (V 114) treats the type-indicating notation 'acb' as an abbreviation for the type-indicating notation '$(ac)b$', so that, according to Church's notation, the type of '$\lambda x\lambda y(\ldots x\ldots y\ldots)$' would be indicated by 'acb' rather than by 'abc'. Similar considerations apply to triple λ-operators, quadruple λ-operators, and so on.

32.26. Sometimes it is useful to assign type to connectives such as '$\sim$', '$\vee$', '$\cdot$', '$\supset$' (06.1). Since '$\sim$' is, in effect, an operator which is applied to expressions of type o to give resulting expressions also of type o, the type of '$\sim$' would be oo. Similarly the types of '$\vee$', '$\cdot$', and '$\supset$' would be ooo. Quantifiers (06.3) are like '$\sim$' in the respect that they are applied to expressions of type o to give resulting expressions also of type o; so that quantifiers would be of type oo. (If quantifiers are defined in terms of operators 'Π' and 'Σ', as indicated in 40.8, the quantifiers would still be of type oo, but 'Π' and 'Σ' would be of type $o(oa)$, where a is the type of the variable involved in the quantifiers.) Type can be assigned to any one-place operator which is restricted to operands of specified type and which gives resulting expressions only of a specified type. Similarly type can be assigned to any n-place operator which is restricted in such a way that a type is specified for each of its n operands, and also a type for the result of its application to the operands. If the respective types of the n operands are required to be types $b_1, b_2, \ldots, b_n$, and if the type of the result of applying the operator is a, then the type of the operator would be $ab_1b_2\ldots b_n$. Unless the λ-operator 'λx' is restricted in the appropriate way, type cannot be assigned to it. If the restriction is made, we have various λ-operators restricted in various ways and having different types. (Cp. Church, V 114.) (If 'λ' itself is viewed as a two-place operator that can be applied to 'x' and '$(\ldots x\ldots)$' to give '$\lambda x(\ldots x\ldots)$', there would also be need to restrict 'λ' appropriately in order to assign type to it, and we would need various such operators, 'λ_a', 'λ_b', and so on, of appropriate types a, b, and so on.)

32.3. Ajdukiewicz has a notation comparable to that of Church, using a fraction '$\frac{a}{b}$' instead of the sequence 'ab'.

32.4. Carnap uses the numerals '0', '1', '2', ..., instead of 'ι', '$o\iota$', '$o(o\iota)$', ..., and he uses 'ab' instead of 'oab' to stand for the type of a binary functional variable of type oab.

32.5. [PM] has no notation to designate a type "absolutely". It uses instead complicated notations to express the type of one variable relative to the type of another (*12).

32.6. Church (XI 31) has proposed systems of type theory that introduce a distinction between sense and denotation (02).

33. Implicit indication of type

33.1. In point of fact, the explicit indication of type by means of subscripts (32.2) can often be avoided since type is sufficiently indicated by context in many situations, at least if the type of a few symbols is already specified.

33.2. For example, suppose that we consider the well-formed formula '$x(y).z(x,y).t(x)$' where the type of 'y' is already specified and where the whole formula is, of course, of type o. If 'y' is of type ι then 'x', 'z', and 't' must be of respective types $o\iota$, $o(o\iota)\iota$, and $o(o\iota)$, while if 'y' is of type $o\iota$, they must be of respective types $o(o\iota)$, $o(o(o\iota))(o\iota)$, and $o(o(o\iota))$.

33.3. Despite the economy that can be introduced by the method of 33.2, it is nevertheless very common to use 'p', 'q', 'r', ..., as propositional variables, 'x', 'y', 'z', ..., as individual variables, and 'F', 'G', 'H', ..., as variables for propositional functions of individuals.

CHAPTER 4

COMBINATORY LOGIC

40. Application and abstraction

40.1. Combinatory logic is concerned with two processes which, in a sense, are inverses of each other. One is the process of finding the value of a function, given the function and an argument of the function. This first process corresponds to *application*, whereby a function is applied to an argument, and the result of the application is the value of the function for that argument. This notion of application is treated as a fundamental primitive concept in combinatory logic, somewhat as addition or multiplication might be treated as primitive in ordinary algebra.

40.2. The second process is the process of finding the function itself, given suitable information about the values of the function for all its arguments. This process corresponds to *abstraction*, and in practice requires that we have at hand an *associated form* of the function (05.4). Thus, if '$(\ldots x\ldots)$' is an associated form of a function, then we have the following information about values of the function for all its arguments: If 'a' denotes any argument of the function, then '$(\ldots a\ldots)$' denotes the corresponding value of the function. By use of the λ-operator (lambda operator) 'λx' containing the same variable 'x' as in the associated form '$(\ldots x\ldots)$', we can form the expression '$\lambda x(\ldots x\ldots)$' which may be regarded as denoting the function itself (cp. 06.2).

40.3. Strictly speaking, there are two kinds of abstraction, one concerned with obtaining a name of a function by operating with the λ-operator on an associated form of the function, the other concerned with obtaining the function itself from suitable information about its values for all its arguments. The first kind of abstraction may be called syntactical abstraction and the second kind real abstraction. These two kinds of

abstraction are parallel in the sense that the function obtained by real abstraction will be denoted by a name (of a function) obtained by syntactical abstraction. (There is also abstraction as the semantical relation of the associated form of a function to the function itself. This third meaning of the term 'abstraction' is the one used by Church ([Ch] 22).)

40.4. Similarly there are two kinds of application, one concerned with obtaining a name of a value of the function from a name of the function and a name of an argument of the function, the other concerned with obtaining the value itself from the function and an argument of the function. These two kinds of application may be called *syntactical application* and *real application* (or simply *application*). They are parallel to each other in the same way that the two kinds of abstraction are parallel to each other. When an operator is "applied" to an operand, this sort of application can be regarded as a somewhat more general notion of syntactical application which we may call *operative application*. In case the operator does not contain any binding variables (06.1), the application of such an operator may be regarded as a case of ordinary syntactical application; but if it contains binding variables, the application of the operator is rather a case of the more general kind of syntactical application, operative application. The syntactical application typical of combinatory logic is called *combination* and is always application of a name of a function to a name of a single argument. It is a case of syntactical application, in the narrower sense. On the other hand, application of the λ-operator is a case of operative application but not of syntactical application in the narrower sense. All systems of combinatory logic use combination, but not all use the λ-operator and operative application. (Church's system of λ-conversion uses both, but the example in 42 uses only combination.)

40.5. In those systems of combinatory logic which do not use the λ-operator, the effect of the λ-operator is achieved indirectly by use of so-called *combinators* (43.7) in terms of which it is possible to define expressions of the form '$[x](\ldots x\ldots)$' (42.6, 45.6). The latter expressions play the role played by expressions of the form '$\lambda x(\ldots x\ldots)$' elsewhere in combinatory logic. It is possible to regard '$[x]$' as a pseudo-operator that is closely analogous to the operator 'λx'. See [CF] 189.

40.6. The concept of abstraction is widely used in symbolic logic outside of combinatory logic. For example, a propositional function may be defined by replacing a constant 'a' by a variable 'x' in a sentence '$(\ldots a\ldots)$', so that the result '$(\ldots x\ldots)$' would be an associated form of the propositional function, and '$\lambda x(\ldots x\ldots)$' would denote the propositional function. More particularly, if '$\sim f(a)$' expresses a proposition to the effect that a is not a member of f, then '$\lambda x(\sim f(x))$' would denote a propositional function, which, when applied to a, has the proposition $\sim f(a)$ as its value. Thus the proposition $(\lambda x(\sim f(x)))(a)$ asserts, in effect, the same thing as $\sim f(a)$. It is to be noted that '$\lambda x(\sim f(x))$' can be regarded as denoting the complement of the class f (54.2). Similarly, '$\lambda x[f(x).g(x)]$' can be regarded as denoting the intersection (54.3) of the classes f and g, and '$\lambda x\lambda y[r(x, y).s(x, y)]$' can be regarded as denoting the relational intersection (60.63) of the relations r and s.

40.7. Outside of combinatory logic (51.2, 60.22), it is usual to employ the notation '$\hat{x}$' instead of 'λx' in the case of propositional functions. Thus '$\hat{x}(\sim f(x))$' would be more commonly used to express the complement of f, rather than '$\lambda x(\sim f(x))$'. In general, '$\hat{x}(\ldots x\ldots)$' is to be viewed as denoting a one-argument propositional function, and hence a class or attribute. Similarly, '$\hat{x}\hat{y}(\ldots x\ldots y\ldots)$' takes the place of '$\lambda x\lambda y(\ldots x\ldots y\ldots)$' and is to be viewed as denoting a two-argument propositional function, and hence a two-term relation or relationship (cp. 05.8, 05.9).

40.8. From the standpoint of combinatory logic, furthermore, a universal statement, '$(x)(\ldots x\ldots)$' (21.1 ff.), can be regarded as expressible as '$\Pi(\lambda x(\ldots x\ldots))$', where '$\Pi$' operates on the class name '$\lambda x(\ldots x\ldots)$' to give a sentence asserting that the class named by '$\lambda x(\ldots x\ldots)$' is universal (all-inclusive). Thus the statement, '$(x)[x=x]$' (meaning that everything is self-identical), may be viewed as equivalent to (or perhaps even as an abbreviation for) the statement, '$\Pi(\lambda x[x=x])$', asserting that the class of self-identical things is all-inclusive (46.2). Similarly an existential statement, '$(\exists x)(\ldots x\ldots)$', can be regarded as expressible as '$\Sigma(\lambda x(\ldots x\ldots))$', where '$\Sigma$' operates on the class name '$\lambda x(\ldots x\ldots)$' to give a sentence asserting that the class named by '$\lambda x(\ldots x\ldots)$' is non-empty. Thus the statement, '$(\exists x)[x=d]$' (meaning that something is identical with d), may be viewed as equivalent to (or perhaps even as an abbreviation for) the statement, '$\Sigma(\lambda x[x=d])$', asserting that the class of things identical with d is non-empty. (The operators 'Π' and 'Σ' correspond to 'A' and 'E' in [F] (128, 145), where they are used without the restrictions of a theory of types. If types are used (31.1 ff.), however, and if 'x' in '$\lambda x(\ldots x\ldots)$' is of type a, then '$\lambda x(\ldots x\ldots)$' is of type oa, so that if 'Π' and 'Σ' operate on expressions of type oa to give expressions of type o, they must be of type $o(oa)$.)

40.9. A descriptive phrase, '$(\imath x)(\ldots x\ldots)$' (26.1 ff.), could, from the standpoint of combinatory logic, be viewed as expressible as '$\imath(\lambda x(\ldots x\ldots))$', where '$\lambda x(\ldots x\ldots)$' denotes a class having only one member and where '$\imath$' (inverted Greek iota) would be an operator which would operate on the class name '$\lambda x(\ldots x\ldots)$' to give a name of the only member of the class denoted by '$\lambda x(\ldots x\ldots)$'. Such an operator, however, does not seem to have been introduced systematically into combinatory logic.

41. The notation of combinatory logic

41.1. In combinatory logic the fundamental method of building complex expressions from simple expressions is the process of *combining* two expressions. The combination of 'a' with 'b' is written '(ab)' or simply 'ab'. This kind of combination is essentially the same as that discussed in 40.4. If 'a' and 'b' are wffs (well-formed formulas) of combinatory logic, so is 'ab'. It is usual to assume that certain capital letters such as 'B', 'C', 'W', 'S', 'K', or others, are the atomic wffs and that all expressions obtained from these by repeating the process of combination are also wffs. (In practice italic rather than roman capital letters are usually used.) For example, if 'B' and 'C' are regarded as atomic wffs, then 'B', 'C', 'BB', 'B(BB)', 'CB', '(CB)C', '((CC)C)B', and so on would be wffs. It is a convention of combinatory logic that we may write 'abc' for '$(ab)c$', and '$abcd$' for '$((ab)c)d$', and so on. For example, we can write 'CBC' for '(CB)C', and we can write 'CCCB' for '((CC)C)B'. Also, we can write 'C(CC)B' for '(C(CC))B', and 'B(W(WW))WW' for '((B(W(WW)))W)W'. But this convention would not permit omission of parentheses in 'C(CC)B' or in 'B(W(WW))W'. In general, only parentheses that "converge to the left" may be omitted. Thus 'B((WW)W)W' could be rewritten as 'B(WWW)W' but *not* as 'BWWWW'. In other words, omitted parentheses are assumed to associate to the left (cp. 16.51).

41 2. In combinatory logic the combination of one wff with another gives a wff which denotes the result of *applying* a function denoted by the first wff to the entity denoted by the second. (The application of the function is real application in the sense of 40.4.) Thus if 'F' denotes a function f and if 'A' denotes something a which can be an argument of f, then 'FA' would denote $f(a)$; that is, it would denote the value of f for the argument a; in other words, it would denote the result of applying f to a (cp. 05.3). The wffs of combinatory logic can all be viewed as denoting functions. In case 'A' in the above example denotes something a that cannot be an argument for f, we can redefine f so that a is permitted as an argument, and we can arbitrarily specify the value $f(a)$ of f for the argument a. Thus, in practice, every wff of combinatory logic can be viewed as denoting a function, and the domain of definition of each of these functions can be regarded as containing all the things denoted by wffs. In particular each function denoted by a wff is itself in its own domain of definition.

41.3. Just as 'FA' denotes the value of the (one-argument) function f for the argument a (where 'F' denotes f and 'A' denotes a), so also in combinatory logic, 'FAB' is viewed as denoting the value of a two-argument function f for the two arguments a and b. In other words, just as 'FA' denotes $f(a)$, so also 'FAB' is simply a rewriting of '$(FA)B$'. Hence we can also regard 'FAB' as denoting $(f(a))(b)$. Thus it is implicit in the notation of combinatory logic that $f(a, b)$ should be regarded as the same as $(f(a))(b)$. This means that functions of two arguments can be viewed as functions of one argument. To evaluate such a function f, we evaluate it first as a one-argument function of its first argument a, and then treat that value, namely $f(a)$, as a one-argument function of the second argument b. The evaluation of this new function, $f(a)$, for the second argument, b, gives $(f(a))(b)$, which is then regarded as the same as $f(a, b)$. In a similar way, '$FABC$' would be regarded as denoting $f(a, b, c)$, and '$FABCD$' would be regarded as denoting $f(a, b, c, d)$, and so on, where 'F', 'A', 'B', 'C', 'D',..., respectively denote $f, a, b, c, d, \ldots$.

41 4. Suppose that f is a propositional function of one argument (05.7) and that a belongs to the domain of definition of f. Then f may be regarded as an attribute or class (05.8), and $f(a)$ may be regarded as the proposition asserting that a has the attribute f (or is in the class f). In combinatory logic, as was noted in 41.3 above, if we let 'F' denote the function f, and if we let 'A' denote an argument a, then 'FA' would denote $f(a)$, which in the case we are now considering is a proposition. Thus, in combinatory logic we use the notation 'FA' to denote the proposition that asserts that a is a member of (or has as an attribute) f. Similarly, if 'G' denotes a two-termed relation g (or relationship g (05.9)), and if 'A' and 'B' respectively denote 'a' and 'b', then 'GAB' would denote the proposition $g(a, b)$ that asserts that a bears the relation g to b. Furthermore, since 'GAB' is an abbreviation for '$(GA)B$', we see that the proposition denoted by 'GAB' can be viewed as asserting that b is a member of a class denoted by 'GA' (just as the proposition denoted by 'FB' asserts that b is a member of f). This class denoted by 'GA' clearly must be the class of things which are borne g by a, and to say that b belongs to this class is logically equivalent to saying that a bears g to b. Thus, in general in combinatory logic, if 'G' denotes a two-termed relation (or relationship) g and if 'A' denotes something a, then 'GA' denotes the class of (or attribute possessed by just those) things that are borne g by a. In particular, the symbol 'Q' is often used in combinatory logic to denote identity (42.2), so that 'QA' would be the class of things borne identity by a, that is, the class of things with which a is identical. Thus 'QA' denotes the unit class having a as its only member (64.2). This class could also be denoted by '$=A$' if identity is denoted by '=' instead of by 'Q'.

41.5. Occasionally the notation '$F(A)$', '$F(A, B)$', '$F(A, B, C)$',..., is used in combinatory logic instead of 'FA', 'FAB', '$FABC$',.... This may be done by using '$\{F\}(A)$' in place of 'FA', and by defining '$F(A)$', '$F(A, B)$', '$F(A, B, C)$',..., respectively as abbreviations for '$\{F\}(A)$', '$\{\{F\}(A)\}(B)$', '$\{\{\{F\}(A)\}(B)\}(C)$',... (cp. Church 359*4*). More simply we may use '$F(A)$' in place of 'FA', and define '$F(A, B)$', '$F(A, B, C)$',..., as '$F(A)(B)$', '$F(A)(B)(C)$',... (cp. Fitch, *Philosophy of Science*, Vol. 25 (1958) pp. 263–279).

42. Example of a simple system of combinatory logic

42.1. In order to clarify the meaning of some of the more common symbols of combinatory logic we now present an example of a simple

system of combinatory logic. The notation used below is essentially standard though the capital letters would normally appear as italic capitals.

42.2. Let 'Q', 'I', 'B', 'C', 'W', and 'K' be the atomic wffs of the system. (We can also introduce variables as further atomic wffs, but variables are not required for this system.) If 'a' and 'b' are wffs of the system, then 'ab' is a wff of the system. We assume the convention stated in 41.1. To say that a wff 'a' is a theorem of the system, we write:

$$\vdash a.$$

We write '$[a = b]$' for 'Qab'. The axioms of the system will consist of all wffs of the following forms:

$[a = a]$	(Axioms of reflexivity of identity)
$[Ia = a]$	(Axioms for 'I')
$[Babc = a(bc)]$	(Axioms for 'B')
$[Cabc = acb]$	(Axioms for 'C')
$[Wab = abb]$	(Axioms for 'W')
$[Kab = a]$	(Axioms for 'K').

The rules of inference of the system would be the following two (plus a tacit rule to the effect that if 'a' is an axiom, then $\vdash a$):

(i) If $\vdash [a = b]$ or $\vdash [b = a]$, and if $\vdash (\ldots a \ldots)$, where '$(\ldots a \ldots)$' is any wff containing the wff 'a', then $\vdash (\ldots b \ldots)$, where '$(\ldots b \ldots)$' is a wff just like '$(\ldots a \ldots)$' except that 'b' has replaced 'a' in one or more places. (Rule of replacement for identity.)

(ii) If $\vdash [ac = bc]$ for every wff 'c', then $\vdash [a = b]$. (Rule of extensionality, sometimes called rule zeta.)

42.3. In this system we can easily show that if $\vdash [a = b]$, then $\vdash [b = a]$; and also: if $\vdash [a = b]$ and $\vdash [b = c]$, then $\vdash [a = c]$.

42.4. In order to show that $\vdash [\mathrm{BI} = \mathrm{I}]$ we could proceed as follows:

(1) $\vdash [\mathrm{BI}ab = \mathrm{I}(ab)]$	(Axiom for 'B')
(2) $\vdash [\mathrm{I}(ab) = ab]$	(Axiom for 'I')
(3) $\vdash [\mathrm{BI}ab = ab]$	(1), (2), (i)
(4) $\vdash [\mathrm{I}a = a]$	(Axiom for 'I')
(5) $\vdash [\mathrm{BI}ab = \mathrm{I}ab]$	(3), (4), (i)
(6) $\vdash [\mathrm{BI}a = \mathrm{I}a]$	(5), (ii)
(7) $\vdash [\mathrm{BI} = \mathrm{I}]$	(6), (ii).

42.5. In this system we can also show, for example, $\vdash [\mathrm{I} = \mathrm{CKK}]$, or more generally, $\vdash [\mathrm{I} = \mathrm{CK}a]$. In fact, the axioms for 'I' could be omitted, since 'I' could be treated as an abbreviation for 'CKK'.

42.6. The importance of the axioms for 'I', 'B', 'C', 'W', and 'K' is seen from the fact that the following result can be obtained: If '$(\ldots a \ldots)$' is any wff in which the wff 'a' occurs one or more times, then there can be found a wff 'f' such that $\vdash [fa = (\ldots a \ldots)]$, and indeed such that, $\vdash [fb = (\ldots b \ldots)]$, where '$(\ldots b \ldots)$' is obtained from '$(\ldots a \ldots)$' by replacing '$a$' by '$b$' throughout '$(\ldots a \ldots)$'. For example, if '$a$' is 'W' and if '$(\ldots a \ldots)$' is 'B(WW)', then the required 'f' is 'W(BB)', and we can show $\vdash [\mathrm{W(BB)W} = \mathrm{B(WW)}]$, and indeed $\vdash [\mathrm{W(BB)}b = \mathrm{B}(bb)]$ for every wff 'b'. The required 'f' could be viewed as denoting the function denoted by '$\lambda x(\ldots x \ldots)$' in the notation of the λ-operator (06.2) and this 'f' could be abbreviated as '$[x](\ldots x \ldots)$' (45.6). Thus '$\lambda x(\mathrm{B}(xx))$' could be viewed as denoting the same function as 'W(BB)'. Thus it is seen that the effect of the use of the λ-operator is indirectly obtainable in the theory of combinators.

42.7. In the notation used by Curry we could write simply

$$a = b$$

instead of

$$\vdash [a = b].$$

43. Combinations and the theory of combinators

43.1. In combinatory logic we may say not only that 'ab' is the combination of 'a' with 'b', but we may also use the term 'combination' in the following way: The class of combinations of wffs 'a_1', 'a_2',..., 'a_n' is defined as follows:

(1) Each of 'a_1', 'a_2',..., 'a_n' is a combination (in a trivial sense) of 'a_1', 'a_2',..., 'a_n'.

(2) If 'a' and 'b' are combinations of 'a_1', 'a_2',..., 'a_n', so is 'ab'.

(3) The only combinations of 'a_1', 'a_2',..., 'a_n' are those which can be shown to be such by use of (1) and (2). For example, 'a_1', 'a_2a_1', '$a_1a_1(a_2a_1)$', and '$a_3a_5(a_2(a_2a_1))$' are combinations of 'a_1', 'a_2', 'a_3', 'a_4', and 'a_5'.

43 2. Suppose that 'b' is a combination of 'a_1', 'a_2',..., 'a_n' and that 'f' has the following property for all 'c_1', 'c_2',..., 'c_n': If 'd' is the result of replacing 'a_1', 'a_2',..., 'a_n' respectively by 'c_1', 'c_2',..., 'c_n' throughout 'b', then

$$fc_1 \dots c_n = d.$$

Such an 'f' is said to be a *proper combinator*. For example, 'W' is a proper combinator, because if 'b' is chosen as '$a_1a_2a_2$', then, for all 'c_1' and 'c_2' we have

$$Wc_1c_2 = c_1c_2c_2.$$

Similarly 'I', 'B', 'C', and 'K' are proper combinators because we have:

$$\begin{aligned} Ic_1 &= c_1, \\ Bc_1c_2c_3 &= c_1(c_2c_3), \\ Cc_1c_2c_3 &= c_1c_3c_2, \\ Kc_1c_2 &= c_1. \end{aligned}$$

Notice the 'Q' is not a proper combinator. (By 43.7 it is not even a combinator.) We say that 2 is the *order* of 'W' and of 'K', that 1 is the order of 'I', and that 3 is the order of 'B' and 'C'. In general, the order of a proper combinator 'f' is the smallest n for which 'f' has the property described above.

43.3. Most systems of combinatory logic contain the proper combinators 'I', 'B', 'C', 'W' (that is, proper combinators having the properties assigned to 'I', 'B', 'C', 'W' in the system of 42) and sometimes also 'K'. These proper combinators might appear as atomic wffs, as in 42.2, or they might be definable as certain combinations of other proper combinators. 'I' is called the *elementary identificator*, 'B' is called *the elementary compositor*, 'C' is called *the elementary permutator*, 'W' is called *the elementary duplicator*, and 'K' is called *the elementary cancellator*.

43.4. Other proper combinators frequently used in combinatory logic are 'S', 'T', 'Φ', 'Ψ', and 'J'. (Sometimes 'T' is written as 'C_*'.) The identities characterizing these combinators are as follows:

$$\begin{aligned} Sabc &= ac(bc), \\ Tab &= ba, \\ \Phi abcd &= a(bd)(cd), \\ \Psi abcd &= a(bc)(bd), \\ Jabcd &= ab(adc). \end{aligned}$$

43.5. There are also so-called *numerical combinators*, '0', '1', '2', '3',..., which are proper combinators having the properties:

$$\begin{aligned} 0ab &= a, \\ 1ab &= ab, \\ 2ab &= a(ab), \\ 3ab &= a(a(ab)), \text{ and so on.} \end{aligned}$$

Sometimes the notation 'Z_0', 'Z_1', 'Z_2',..., is used in place of '0', '1', '2',....

43.6. If the numerical combinators are treated as denoting the non-negative integers, then the proper combinator 'B' may be regarded as denoting multiplication, and we may write '$[a \times b]$' or '$(a \cdot b)$' as an abbreviation for 'Bab'. Furthermore, the proper combinator 'ΦB' may be regarded as denoting addition, and we may write '$[a+b]$' or '$(a+b)$' as an abbreviation for 'ΦBab'. The identity characterizing 'ΦB' is the following:

$$\Phi Babcd = ac(bcd).$$

Exponentiation can be handled easily by defining 'a^b' as an abbreviation for 'ba'.

43.7. A *combinator* is defined as any combination of proper combinators. Some combinations of proper combinators are themselves proper combinators, others are not. For example, 'ΦB' is a combination of the proper combinators 'Φ' and 'B' and is a proper combinator. Similarly 'CI' is a proper combinator, but 'WWW' and 'CII' are not proper combinators, though they are combinations of proper combinators. (Curry and Feys [CF] treat combinators as the functions denoted by the expressions here called combinators.)

43 8. Some combinators which are not proper combinators represent important operations. For example, consider the combinator 'Υ' defined as 'WS(BWB)' (This Greek symbol is the capital Greek letter upsilon.) This combinator is not proper. It has the property

$$\Upsilon a = a(\Upsilon a).$$

We can show this as follows:

$$\begin{aligned}
\Upsilon a &= \mathrm{WS(BWB)}\, a,\\
\mathrm{WS(BWB)}\, a &= \mathrm{S(BWB)}\, aa,\\
\mathrm{S(BWB)}\, aa &= \mathrm{BWB}a(\mathrm{BWB}a),\\
\mathrm{BWB}a(\mathrm{BWB}a) &= \mathrm{W}(\mathrm{B}a)(\mathrm{BWB}a),\\
\mathrm{W}(\mathrm{B}a)(\mathrm{BWB}a) &= \mathrm{B}a(\mathrm{BWB}a)(\mathrm{BWB}a),\\
\mathrm{B}a(\mathrm{BWB}a)(\mathrm{BWB}a) &= a(\mathrm{BWB}a(\mathrm{BWB}a)),\\
a(\mathrm{BWB}a(\mathrm{BWB}a)) &= a(\Upsilon a).
\end{aligned}$$

This combinator is important in connection with the Russell paradox ([CF] 258–262) and in connection with recursion and mathematical induction. (Closely similar operators are used in Fitch XX 81 and XXIV 240.) This combinator also is important in connection with combinatory analysis of the feedback aspects of sequential circuits. (It is the same as the operator 'Z_1' in Fitch, "Representation of Sequential Circuits in Combinatory Logic", *Philosophy of Science*, Vol. 25 (1958) pp. 263–279.)

44. Lambda-conversion

44.1. Instead of starting with undefined combinators such as 'I', 'B', 'C', 'W', Church has developed a kind of combinatory logic that starts with the concept of functional abstraction as embodied in the use of the λ-operator (cp. 06.2). Here variables play an important role.

44.2. Infinitely many symbols are specified as being variables of the system. The class of wffs (well-formed formulas) of a system of *λ-conversion* (*lambda-conversion*) is definable as follows:

(1) Every variable is a wff.
(2) If 'a' and 'b' are wffs, so is 'ab' (cp. 41.1).
(3) If 'x' is a variable and if 'a' is a wff, then 'λxa' is a wff. (Some definitions of the class of wffs might require here that 'x' occur free in 'a'.)
(4) All wffs are such solely in virtue of (1)–(3).

44.3. 'λxa' may also be written as '$\lambda x(a)$' or as '$\lambda x.a$'. If 'a' can be viewed as an associated form of a function (cp. 05.4) with respect to the variable 'x', then '$\lambda x.a$' can be viewed as denoting the function itself. For example, '$\lambda x.x^2$' would denote a function f such that $f(x) = x^2$. (Here we can think of 'x^2' as defined as at the end of 43.6.)

44.4. If 'x' is a variable and if 'a' is a wff, then '$\lambda x.a$' may be called an *abstract* and 'λx' is an *abstractor*. The abstractor 'λx' is a one-place operator (06.1). In conformity with 06.4 and 21.6, it can be said that an occurrence of a variable 'x' in a wff 'a' is a bound occurrence of 'x' with respect to (and in) 'a' if the occurrence in question is in a well-formed part of 'a' of the form '$\lambda x.b$'; otherwise the occurrence of 'x' is said to be a free occurrence of 'x' with respect to (and in) 'a'. More generally, an occurrence of a wff 'c' in a wff 'a' is said to be a bound occurrence of 'c' in 'a' if some occurrence of a variable that is free in 'c' is bound in 'a'; otherwise the occurrence of 'c' in 'a' is said to be free in 'a'.

44.5. '$\lambda x\lambda y.a$' is an abbreviation for '$\lambda x(\lambda y.a)$', and '$\lambda x\lambda y\lambda z.a$' is an abbreviation for '$\lambda x(\lambda y(\lambda z.a))$', and so on. Further notational economy is obtained by the abbreviations: '$\lambda xy.a$' for '$\lambda x\lambda y.a$'; '$\lambda xyz.a$' for '$\lambda x\lambda y\lambda z.a$' and so on.

44.6. Church defines a relation of *interconvertibility* or *conversion* between wffs of his system of λ-conversion in such a way that interconvertible wffs can be thought of as denoting identical (or, in some respect, equivalent) concepts. This notion of conversion or interconvertibility in Church's systems plays a role very similar to that played by the relation which Curry denotes by '='. In fact the theory of combinators and the theory of λ-conversion are, essentially, slightly different ways of dealing with the same subject-matter, and the fundamental concepts of each theory are definable (with minor qualifications) within the other.

44.7. Church's three *conversion rules* (*rules of λ-conversion*) may be paraphrased as follows, combining his second and third rules into a single rule:

I. Wffs 'a' and 'b' are convertible into each other provided that they are just alike except that a well-formed part of 'b' differs from the corre-

sponding well-formed part of 'a' in having all bound occurrences of a variable 'x' replaced by bound occurrences of a variable 'y'. (It is to be understood that the occurrences of 'x' are bound with respect to the part of 'a' in question, and that the occurrences of 'y' are bound with respect to the corresponding part of 'b'. It is also to be understood that 'x' and 'y' do not both occur bound in the part of 'a' in question. The "parts" of 'a' and 'b' in question might be 'a' and 'b' themselves, and similarly in Rule II.)

II. Wffs 'a' and 'b' are convertible into each other provided that they are just alike except that a well-formed part of 'a' is of the form '$(\lambda x.c)d$', while the corresponding well-formed part 'e' of 'b' is the result of replacing in 'c' the occurrences of 'x' that are free (in 'c') by occurrences of 'd' that are free in 'e'.

44.8. Rule I above corresponds to what Curry calls the rule of "α-conversion" while Rule II corresponds to what he calls the rules of "β-conversion". The following rule corresponds to a rule called by Curry the rule of "η-conversion":

> Wffs 'a' and 'b' are convertible into each other provided that they are just alike except that a well-formed part of 'a' is of the form '$(\lambda x.cx)$', where 'x' is not free in 'c', while the corresponding part of 'b' is 'c' itself.

44.9. The relation of convertibility is assumed to be reflexive, symmetrical, and transitive, as well as to satisfy rules I and II (and perhaps also the rule of η-conversion). To say that 'a' is convertible into 'b', we may write 'a cnv b'. The following are examples of valid conversions, assuming 'x' and 'y' are distinct variables that do not occur free in 'f':

$(\lambda x.fx)a$ cnv $(\lambda y.fy)a$	(by α-conversion),
$\lambda x.fx$ cnv f	(by η-conversion),
$(\lambda x.fx)a$ cnv fa	(by β-conversion or by η-conversion),
$(\lambda x.fxy)a$ cnv fay	(by β-conversion),
$(\lambda y.(\lambda x.fxy))a$ cnv $\lambda x.fxa$	(by β-conversion),
$(\lambda yx.fxy)a$ cnv $\lambda x.fxa$	(by β-conversion; cp. 44.5),
$(\lambda yx.fxy)ab$ cnv $(\lambda x.fxa)b$	(by β-conversion),
$(\lambda x.fxa)b$ cnv fba	(by β-conversion),

$(\lambda yx.fxy)ab$ cnv fba	(by β-conversion twice and transitivity of conversion).

Similarly these hold in the reverse direction, for example

$$fa \text{ cnv } (\lambda x.fx)a.$$

45. Definitions of combinators as abstracts

45.1. The proper combinators 'I', 'C', 'W', 'B', 'S', 'T', 'Φ', 'Ψ' and 'J' may be defined in Church's system of λ-conversion by means of the following abbreviations, where 'x', 'y', 'z' and 'w' are distinct variables:

'I' for '$\lambda x.x$',
'C' for '$\lambda xyz.xzy$',
'W' for '$\lambda xy.xyy$',
'B' for '$\lambda xyz.x(yz)$',
'S' for '$\lambda xyz.xz(yz)$',
'T' for '$\lambda xy.yx$',
'Φ' for '$\lambda xyzw.x(yw)(zw)$',
'Ψ' for '$\lambda xyzw.x(yz)(yw)$',
'J' for '$\lambda xyzw.xy(xwz)$'.

45.2. As an example of the use of the above definitions, we can show that $Babc$ cnv $a(bc)$. This corresponds to showing that $Babc = a(bc)$ in Curry's theory of combinators. The procedure may be outlined as follows: We first rewrite '$Babc$' as

$$(\lambda xyz.x(yz))abc.$$

We assume that 'x', 'y', and 'z' have been chosen in such a way that they do not occur free in 'a', 'b', and 'c'. Then, in accordance with 44.5 we rewrite the above formula thus:

$$(\lambda x(\lambda y(\lambda z.x(yz))))abc.$$

Then we have:

$(\lambda x(\lambda y(\lambda z.x(yz))))abc$ cnv $(\lambda y(\lambda z.a(yz)))bc$,
$(\lambda y(\lambda z.a(yz)))bc$ cnv $(\lambda z.a(bz))c$,
$(\lambda z.a(bz))c$ cnv $a(bc)$.

45.3. If the class of wffs is defined in such a way that '$\lambda x.a$' is a wff if 'x' is a variable and 'a' is a wff, regardless of whether or not 'x' occurs free in 'a' (see (3) of 44.2), then the combinator 'K' may be defined as an abbreviation for the following abstract:

$$\lambda xy.x.$$

To show that Kab cnv a, we rewrite '$\lambda xy.x$' as '$\lambda x(\lambda y.x)$', and then we have (assuming, of course, that 'x' and 'y' are not free in 'a'):

$$(\lambda x(\lambda y.x))ab \text{ cnv } (\lambda y.a)b,$$
$$(\lambda y.a)b \text{ cnv } a.$$

45.4. If the class of wffs has been defined as indicated in 45.3, then the numerical combinators may be defined by way of the following abbreviations, where 'x' and 'y' are distinct variables:

'0' for '$\lambda xy.y$',
'1' for '$\lambda xy.xy$',
'2' for '$\lambda xy.x(xy)$',
'3' for '$\lambda xy.x(x(xy))$',
'4' for '$\lambda xy.x(x(x(xy)))$', and so on.

The stipulation at the beginning of 45.3 is required in connection with the definition of '0' but not in connection with the definition of the other numerical combinators.

45.5. The combinator denoting multiplication may be defined in the same way as 'B', since we can regard 'B' as denoting multiplication. The combinator '+' for addition may be defined as '$\lambda xyzw.xz(yzw)$' or as '$\lambda xyz.\mathrm{B}(xz)(yz)$'. If we write '$[a \times b]$' for '$\mathrm{B}ab$' and '$c^{[a]}$' for '$dc$', then '+' could be defined as '$\lambda xyz.[z^x \times z^y]$'. (See also 43.6.) We can then show that $+abc$ cnv $[c^a \times c^b]$. If we write '$[a+b]$' for '$+ab$' and '$c^{[a+b]}$' for '$[a+b]c$', the above conversion becomes:

$$c^{[a+b]} \text{ cnv } [c^a \times c^b].$$

45.6. Just as individual combinators can be defined by use of the λ-operator, so also the λ-operator, or at least an operation essentially equivalent to it, can be defined in terms of combinators. (Cp. remarks at the end of 42.6.) This operator, so defined, is written as '$[x]$' by Curry,

where 'x' is a variable. Thus in the combinatory system of section 42, the operator '$[x]$' could be suitably defined and could be used to play a role similar to that played by 'λx' in systems of λ-conversion. It might be convenient, in this connection, to assume that the system includes variables, but it is not necessary to do so. The variables that appeared in operators of the form '$[x]$' and in expressions of the form '$[x](\ldots x\ldots)$' could be treated as eliminable by definition. (For some methods for defining '$[x]$' see [CF], pp. 188 ff.)

46. Further extensions of combinatory logic

46.1. Combinatory logic has been extended considerably beyond the theory of combinators (cp. [CF] pp. 257 ff.). In particular, Curry has introduced an operator 'F' such that if 'a' and 'b' denote categories and if 'g' denotes a function, then '$\mathrm{F}abg$' may be interpreted as meaning that, for all c, if c is in category a, then $g(c)$ is in category b. (Here we should write 'gc' instead of '$g(c)$' if we wish to conform fully to combinatory notation.) The theory of 'F' developed by Curry is called the "theory of functionality".

46.2. In connection with the theory of functionality, the operator 'Π' of unlimited universality and 'Ξ' of restricted universality are considered. The expression 'Πa' may be thought of as roughly equivalent to '$(x)(ax)$' (that is, to the statement that a is a universal class), and the expression 'Ξab' may be thought of as roughly equivalent to '$(x)[ax \supset bx]$' (that is, to the statement that every member of a is a member of b).

CHAPTER 5

CALCULUS OF CLASSES

50. Introduction

50.1. The statement, 'Every man is mortal', can be viewed as asserting that whatever is a man is mortal, or, equivalently, as asserting that the class of men is a subclass of the class of mortals. The former viewpoint is appropriate in connection with the first-order functional calculus, since we may then express the statement as '$(x)[f(x) \supset g(x)]$'. The latter viewpoint is appropriate in connection with the calculus of classes, since the statement is then treated as asserting that a certain relation holds between two classes. Indeed, as has elsewhere (05.8) been noted, classes may be thought of as properties (attributes) of a special kind, and properties themselves may be thought of as one-argument propositional functions (05.7).

50.2. The term *set* is often used as synonymous with the term *class*, and the former term could properly replace the latter in many places in this Dictionary. For the sake of uniformity, however, we shall use the term *class* in preference to the term *set* except in special contexts in which these terms have a technical difference in meaning. One of these special contexts is that of so-called *set theory* or *theory of sets*. This field of study provides axiomatic systems for dealing with sets as classes of a special sort, and sometimes makes a sharp technical distinction between the term *set* and the term *class* (82.1). (Cp. Gödel, VI 112.) We will regard this field as somewhat outside the scope of this Dictionary, though the notations of set theory are usually similar to those of the calculus of classes.

50.3. In [Q] (158 ff.) a distinction, based on *stratification* (31.4), is made between those classes that are capable of being *elements* (members) of

classes, and those that are not. This distinction between classes having elementhood and those that do not is somewhat like the distinction between set and class in set theory (cp. 82.1.)

50.4. In the first edition of [PM], which adopts the ramified (branched) theory of types (31.3), classes are represented by class expressions which are defined only contextually (just as definite descriptive phrases, in [PM], are defined only contextually (26.43)). The *Axiom of Reducibility* is required for the adequacy of this treatment of classes. This axiom asserts that corresponding to any property having a type of order $n+m$ (cp. 31.3) and applying to things of a type of order n, there is an equivalent property (applying to the same things, and only those, as the original property) but having a type of order $n+1$. Such a property, of order only one greater than the order of what it applies to, is called by Russell a *predicative property*. According to Russell's theory of classes, any statement ostensibly about the class of things having a property F is really a statement about a predicative property equivalent to F (and by the Axiom of Reducibility there always is such a predicative property). Thus if '$(\ldots x\ldots)$' is a form (05.2) associated with F, and if '$\hat{x}(\ldots x\ldots)$' is a class expression denoting the class of things having F, then a statement, '$(\text{---}\hat{x}(\ldots x\ldots)\text{---})$', about $\hat{x}(\ldots x\ldots)$, can be regarded as actually '$(\exists g)[(x)[g(x) \equiv (\ldots x\ldots)]. (\text{---}g\text{---})]$', where '$g$' is a variable for a predicative property of suitable type. In particular, the order of the type of 'g' would be one greater than the order of the type of the variable 'x'. Russell's own notation ([PM] *20.01) would make use of

'$\phi!\hat{z}$'	in place of 'g',
'$\phi!x$'	in place of '$g(x)$',
'$(\exists\phi)$'	in place of '$(\exists g)$',
'ψx'	in place of '$(\ldots x\ldots)$',
'$f[\hat{z}(\psi z)]$'	in place of '$(\text{---}\hat{x}(\ldots x\ldots)\text{---})$', and
'$\phi!x.\equiv_x.\psi x$'	in place of '$(x)[g(x) \equiv (\ldots x\ldots)]$',

so that, in this notation, he is treating '$f[\hat{z}(\psi z)]$' as an abbreviation for '$(\exists\phi):\phi!x.\equiv_x.\psi x: f(\phi!\hat{z})$'. The presence of the exclamation point '!' indicates that 'ϕ' is a variable for a predicative property This theory of classes purports to give the desired result that classes having the same

members are identical (05.8). In the second edition of *Principia Mathematica*, this whole approach was dropped in favor of the simple theory of types, and classes were identified with propositional functions, but the necessary changes in the system were only roughly sketched. One disadvantage of this revised standpoint is that the simple theory of types provides in itself no solution of the semantical paradoxes, such as Richard's Paradox (3597). (Also the simple theory of types must be supplemented by an axiom of extensionality to guarantee that classes that have the same members are identical.)

51. Variables, abstracts, and constants for classes

51.1. Most systems that deal with classes use special variables which range over classes. For example, [PM] uses 'α', 'β', and 'γ', though there is apparently a confusion in [PM] between such variables and variables that serve metalinguistic purposes (08.3–08.8), as, for example, do the letters 'F', 'G', 'H' in [F]. Actually in [PM] the Greek symbols 'α', 'β', and 'γ' seem to be used both as variables for classes and as metalinguistic variables referring to class expressions.

51.2. Names of classes may be formed, in many systems, by placing a capped variable '$\hat{x}$' to the left of a form '$(\ldots x\ldots)$', giving an expression '$\hat{x}(\ldots x\ldots)$' called an *abstract* (40.7). This notation appears in [PM], [R], and [Q]. (In [PM] the notations '$\phi\hat{x}$' and '$\psi\hat{x}$' are used to refer to propositional functions rather than to classes, and the notations '$\phi!\hat{x}$' and '$\psi!\hat{x}$' refer to predicative propositional functions (cp. 50.4).)

51.3. Other notations for '$\hat{x}(\ldots x\ldots)$' are '$x\in(\ldots x\ldots)$' (Peano and some papers of Quine II 51, III 53), and '$(x\mid(\ldots x\ldots))$' in [F] (94)). In [R] (220) the notations '$\{x\mid(\ldots x\ldots)\}$' and '$\mathrm{E}_x(\ldots x\ldots)$' are mentioned as alternatives to '$\hat{x}(\ldots x\ldots)$'. In [R] (222) there is also a further generalization of this notation so that, for example, '$\{y_1+y_2 \mid y_1\in\alpha\,.\,y_2\in\beta\}$' would denote the class of sums whose first summand is in the class referred to by 'α' and whose second summand is in the class referred to by 'β'. In general, '$\{A\mid P\}$' is treated as an abbreviation for '$\hat{x}(\exists y_1)(\exists y_2)\ldots(\exists y_n)[[x=A]\,.\,P]$' where '$x$' occurs free in neither 'A' nor 'P' and where 'y_1', 'y_2', ..., 'y_n' are all the variables occurring free in either 'A' or 'P'.

51.4. Two important notations are 'V', as used for a name of the *universal class*, and 'Λ', as used for a name of the empty class ([PM] *24). These symbols, being proper names rather than variables, may be called constants (03.1). Sometimes '1' is used to denote the universal class and '0' to denote the empty class. By the universal class is often meant the class of all individuals (05.7, 20.1), and in this case 'V' can be defined as '$\hat{x}[x=x]$', where 'x' is a variable ranging over individuals ([R] 226). Another possible definition of 'V' is '$\hat{x}\,\mathrm{V}$' (cp. 12.7). Where a theory of types is used, there may be a universal class corresponding to each type. In [F] (105) 'U' stands for the universal class; but this class U is the class of all entities and not just of all individuals, and no theory of types is involved. The empty class 'Λ' can be defined as '$\hat{x}[x\neq x]$' ([R] 226) or as '$\hat{x}\,\Lambda$' (cp. 12.7). If the 'x' is a variable ranging over only individuals, then 'Λ' should, in this case, be regarded as denoting the empty class of individuals (05.7, 20.1). Where a theory of types is used, there may be an empty class corresponding to each type. Sometimes the term *null class* is used instead of the term *empty class*. Occasionally a distinction between these two terms is made, so that every "null" class is an "empty" class, but only those "empty" classes that are necessarily "empty" are "null".

51.5. If a theory of types is used in dealing with classes, then types will be assigned to classes in essentially the same way as to propositional functions, since classes can be viewed as propositional functions (or attributes) that satisfy an extensionality principle (05.8). Consequently, abstracts and other notations for classes may involve indications of types (32).

52. Operators corresponding to those of the functional calculus

52.1. The symbol '$\in$' may be used in such a way that '$x\in\alpha$' is definable as, or roughly equivalent to, '$\alpha(x)$' (cp. the functional notation in 05.3 and 20.3; also [F] (89)). '$x\in\alpha$' may be taken to mean that x is a member of the class α. In [Q] (162) '$x\in\mathrm{V}$' is taken to mean that x is an "element". When types are used, and when '$\in$' is in the context '$x\in\alpha$' where 'x' is of type ι, the symbol '$\in$' is of type $o\iota(o\iota)$ (cp. 32.21–32.24). The notation '$x, y\in\alpha$' is often used as an abbreviation for '$[x\in\alpha]\,.\,[y\in\alpha]$' ([PM] *20.04). (In the foregoing sentences and hereafter, the Greek letters 'α', 'β', and

'γ' are used like italic letters for metalinguistic purposes. See 08.3–08.8 and especially 08.7.)

52.2. If 'α' is '$\lambda x(\ldots x\ldots)$' or '$\hat{x}(\ldots x\ldots)$', then '$(x)(\ldots x\ldots)$' may be treated as '$\Pi\alpha$', where '$\Pi$' is an operator of combinatory logic (40.8) and '$\Pi\alpha$' means that α is a universal class. Similarly, [F] (132) treats '$(x)[x\in F]$' as an abbreviation for '$F\in A$', meaning that F has the attribute A of universality, or in other words that everything has the attribute F (everything belongs to the class F).

52.3. If 'α' is '$\lambda x(\ldots x\ldots)$' or '$\hat{x}(\ldots x\ldots)$', then '$(\exists x)(\ldots x\ldots)$' may be treated as '$\Sigma\alpha$', where '$\Sigma$' is an operator of combinatory logic (40.8) and '$\Sigma\alpha$' means that α is a non-empty class. The corresponding notation in [PM] *24 is '$\exists!\alpha$'. Similarly, [F] (145) treats '$(\exists x)[x\in F]$' as an abbreviation for '$F\in E$', meaning that F has the attribute E of non-emptiness, or in other words that something has the attribute F (something belongs to the class F).

53. Functions from classes to propositions

53.1. We now consider some functions from classes to propositions, that is, some functions whose arguments are classes and whose values are propositions.

53.2. *Inclusion* is often expressed by the symbol '$\subset$'. The expression '$\alpha\subset\beta$' may be treated as an abbreviation for '$(x)[x\in\alpha\supset x\in\beta]$' ([Q] 185) and may be taken to mean that α is included in β, or that, for every x, if x is an α then x is a β, or, in other words, that α is a subclass of β.

53.3. The notation '$\subseteq$' or '$\subseteqq$' is often used instead of '$\subset$', for example by [R] (234) and [F] (184). The notation '$\alpha\subset\beta$' is then to be taken as meaning that α is included in β but that β is not included in α, in other words, that α is *properly included* in β (is a *proper subclass* of β). In this sense of '$\alpha\subset\beta$', there would have to be some member of β which is not a member of α. (See also 66.05.)

53.4. The symbol '$=$' has already been employed (25) for expressing identity between individuals. This symbol is often also used to express *equality of classes*, that is, *mutual inclusion of classes*. The notation '$\alpha=\beta$' may, therefore, be treated as an abbreviation for '$(x)[x\in\alpha\equiv x\in\beta]$' ([R] 211) and taken to mean that, for every x, x is an α if and only if x is a β. (In [F] (188) the notation '$F\simeq G$' is equivalent to '$(x)[x\in F\equiv x\in G]$'.) On the other hand, this same symbol '$=$' may also sometimes be used to express Leibnizian identity between classes, since the Leibnizian definition of identity may be employed not only for individuals (30.8) but also for classes. If a theory of types is used, this will require a modification of the definition to conform to type requirements. Thus, according to the Leibnizian definition of identity, '$\alpha=\beta$' may be treated as an abbreviation for '$(\gamma)[\alpha\in\gamma\equiv\beta\in\gamma]$', where '$\gamma$' is of appropriately higher type than 'α' and 'β'. Either of these two interpretations of '$\alpha=\beta$' (mutual inclusion of α and β, Leibnizian identity of α and β) may be used, and they are in fact equivalent to each other if, as is usual in dealing with classes, the following principle of extensionality for classes is assumed:

$$(\gamma)[\alpha\in\gamma\equiv\beta\in\gamma]\equiv(x)[x\in\alpha\equiv x\in\beta].$$

This principle asserts, in effect, that two classes are identical (in the Leibnizian sense) if and only if they have the same members. Actually the following weaker principle of extensionality, in most systems of logic, produces the same effect as the above stronger principle:

$$(x)[x\in\alpha\equiv x\in\beta]\supset(\gamma)[\alpha\in\gamma\supset\beta\in\gamma].$$

53.5. '$\alpha\neq\beta$' is frequently used as an abbreviation for '$\sim[\alpha=\beta]$' ([R] 163).

53.6. '$\exists!\alpha$' is used as an abbreviation for '$(\exists x)[x\in\alpha]$' ([PM] *24.03). Thus '$\exists!\alpha$' means that α is non-empty (52.3).

54. Functions from classes to classes

54.1. We next consider some functions from classes to classes, that is, some functions whose arguments and values are classes.

54.2. The class of all things which are not members of a class α is called the *complement* of α and is usually expressed by '$-\alpha$'. (Often comple-

mentation is viewed as relative to some larger class β, so that, relatively to β, the complement of α is the class of all members of β which are not members of α.) Complementation is thus an operation on classes and is usually denoted by '$-$'. If '$-$' is viewed as denoting a function, we may say that the value of the function, for a given class as argument, is the complement of that class. Frequently '$-\alpha$' is treated as an abbreviation for '$\hat{x}(\sim[x \in \alpha])$'. Other alternative notations for '$-\alpha$' are '$\bar{\alpha}$' ([Q] 180, [R] 226) and '$C(\alpha)$' ([R] 226).

54.3. The class of all things each of which is a member of both the classes α and β is called the *intersection* (or *logical product*) of α with β, and is usually expressed by '$\alpha \cap \beta$'. Frequently '$\alpha \cap \beta$' is treated as an abbreviation for '$\hat{x}[x \in \alpha . x \in \beta]$'. The notation '$\alpha . \beta$' or '$\alpha\beta$' is often used to represent the logical product of α with β. The intersection of three classes α, β, and γ can be expressed as '$\alpha \cap \beta \cap \gamma$' and defined either as '$[\alpha \cap \beta] \cap \gamma$' or as '$\alpha \cap [\beta \cap \gamma]$'. Similarly for more than three classes.

54.4. The class of all things each of which is a member of either or both of the classes α and β is called the *union* (or *join* or *logical sum*) of α with β, and is usually expressed by '$\alpha \cup \beta$'. Frequently '$\alpha \cup \beta$' is treated as an abbreviation for '$\hat{x}[x \in \alpha \vee x \in \beta]$'. The notation '$\alpha + \beta$' is often used to express the logical sum of α with β. The union of three classes α, β, and γ can be expressed as '$\alpha \cup \beta \cup \gamma$' and defined either as '$[\alpha \cup \beta] \cup \gamma$' or as '$\alpha \cup [\beta \cup \gamma]$'. Similarly for more than three classes.

54.5. Other functions occasionally used

54.51. The class of those members of α that are not members of β is called the *difference* of α and β, and is usually expressed by '$\alpha - \beta$'. Frequently '$\alpha - \beta$' is regarded as an abbreviation for '$\alpha \cap -\beta$' ([R] 226).

54.52. The class of those members of α that are not members of β and those members of β that are not members of α is called the *symmetric difference* of α and β and is usually expressed by '$\alpha * \beta$'. Frequently '$\alpha * \beta$' is regarded as an abbreviation for '$[\alpha - \beta] \cup [\beta - \alpha]$' ([R] 238). Other notations sometimes used for this class are '$\alpha \oplus \beta$' and '$\alpha \triangle \beta$'. When the notation '$\alpha \oplus \beta$' is used, the symmetric difference of α and β is often called the "circle sum" of α and β.

54.53. The class of those things which are members of the complement of α or members of β is usually expressed by '$\alpha \supset \beta$'. Frequently '$\alpha \supset \beta$' is regarded as an abbreviation for '$-\alpha \cup \beta$' (Quine's 458*5*). (Note the analogy with the use of the sign '$\supset$' as denoting material implication in 12.4, where '$p \supset q$' is definable as '$\sim p \vee q$'.)

54.6. If use is made of ordered couples (60.3) it is possible to define the *Cartesian product*, $\alpha \times \beta$, of class α with class β as the class of those ordered couples $\langle x, y \rangle$ such that x is a member of α and y is a member of β. We can treat '$\alpha \times \beta$' as an abbreviation for '$\hat{z}(\exists x, y)[x \in \alpha . y \in \beta . z = \langle x, y \rangle]$'. The Cartesian product is a function from classes to classes of ordered couples. (See also 62.2.) Sometimes the Cartesian product is called the *cross product* or *direct product*.

55. Functions from classes of classes to classes

55.1. If γ is a class of classes (a class whose members are themselves classes), than the *intersection* of γ (or *logical product* of γ) is defined as the class of those things each of which is a member of every member of γ. The intersection of γ is designated by '$\bigcap\gamma$' in [R] 239 and by '$\mathfrak{D}\gamma$' in [K] 16. It is necessary to distinguish the intersection of a class of classes, as described here, from the intersection of two or more classes as described in 54.3. But, provided that γ is finite, the intersection, $\bigcap\gamma$, of γ in the present sense, can be viewed as the same as the intersection, $\alpha_1 \cap \alpha_2 \cap \alpha_3 \cap \ldots$, of $\alpha_1, \alpha_2, \alpha_3, \ldots$, in the sense of 54.3, where $\alpha_1, \alpha_2, \alpha_3, \ldots$, are all the members of γ. If γ has infinitely many members, there is no intersection of these members in the sense of 54.3, but there is still the intersection $\bigcap\gamma$ in the present sense.

55.2. If γ is a class of classes, then the *union* of γ (or *join*, or *logical sum*, of γ) is defined as the class of all the members of members of γ. The union of γ is designated by '$\bigcup\gamma$' in [R] 259 and by '$\mathfrak{S}\gamma$' in [K] 16. It is necessary to distinguish the union of a class of classes, as described here, from the union of two or more classes as described in 54.4. But, provided that γ is finite, the union, $\bigcup\gamma$, of γ in the present sense, can be viewed as the same as the union, $\alpha_1 \cup \alpha_2 \cup \alpha_3 \cup \ldots$, of $\alpha_1, \alpha_2, \alpha_3, \ldots$, in the sense 54.4, where $\alpha_1, \alpha_2, \alpha_3, \ldots$, are all the members of γ. If γ has infinitely many members,

then there is no union of these members in the sense of 54.4, but there is still the union $\bigcup\gamma$ in the present sense.

55.3. Sometimes the notation '$\prod_{\alpha\in\gamma}\alpha$' is used instead of '$\bigcap\gamma$', and the notation '$\sum_{\alpha\in\gamma}\alpha$' is used instead of '$\bigcup\gamma$'. If f is a function from members of β to classes, and if γ has as members just those classes $f(x)$ such that x is a member of β, then the notation '$\prod_{x\in\beta} f(x)$' or '$\Lambda_{x\in\beta} f(x)$' is sometimes used instead of '$\bigcap\gamma$' and the notation '$\sum_{x\in\beta} f(x)$' or '$V_{x\in\beta} f(x)$' is sometimes used instead of '$\bigcup\gamma$'.

55.4. The relation p (*logical product of* or *intersection of*) may be defined as $\hat{\alpha}\hat{\gamma}[\alpha = \hat{x}(\beta)[\beta\in\gamma \supset x\in\beta]]$, and the intersection $\bigcap\gamma$ may then be viewed as the class which bears the relation p to γ, that is, as $p^{\prime\prime}\gamma$ (where the notation is that of 64.43 and of [PM] *30). This procedure is used in [PM] *40. Similarly, the relation s (*logical sum of* or *union of*) may be defined as $\hat{\alpha}\hat{\gamma}[\alpha = \hat{x}(\exists\beta)[\beta\in\gamma . x\in\beta]]$, and the union $\bigcup\gamma$ may then be viewed as the class which bears the relation s to γ, that is, as $s^{\prime\prime}\gamma$. (The letters 'p' and 's' are not serving as metavariables in this paragraph but are the actual notation used in [PM].)

56. Calculus of individuals

56.1. The calculus of individuals, formulated first in Leśniewski's mereology (see Sobociński XVIII 331), is analogous in some respects to the calculus of classes. This calculus deals with a part-whole relation between individuals, but has no null element (that is, no element analogous to the empty class (51.4)). We may start from an undefined relation of *discreteness*, in terms of which the part-whole relation and the relation of overlapping are defined. The relation of discreteness between individuals is analogous to the relation of mutual exclusion between classes, the part-whole relation between individuals is analogous to the relation of inclusion between classes, and the relation of overlapping between individuals is analogous to the relation of overlapping between classes.

56.2. '$x \overline{\frown} y$' means that x and y are discrete (have no part in common).

56.3. '$x \circ y$' is defined as '$\sim[x \overline{\frown} y]$' and means that x overlaps y. That is, x and y are said to overlap if they have a part in common.

56.4. '$x < y$' is defined as '$(z)[z \overline{\frown} y \supset z \overline{\frown} x]$' and means that x is part of y. That is, x is a part of y if whatever is discrete from y is also discrete from x.

56.5. '$x \lll y$' is defined as '$x < y . x \neq y$' and means that x is a proper part of y. That is, $x \lll y$ if x is a part of y and x is not identical with y.

56.6. The *sum-individual* or *fusion* of a class of individuals is analogous to the logical sum of a class of classes (55.2). The individual which is the fusion of a class α of individuals is the (smallest) individual which has all the members of α as parts. Such a fusion or sum-individual need not be continuous. Each man may be considered to be a part of the sum-individual which is humanity. The universe U is the fusion of the class of all individuals.

56.7. The *product-individual* or *nucleus* of a class of individuals is analogous to the logical product of a class of classes (55.1). The individual (if any) which is the nucleus of a class α of individuals is the (largest) individual which is part of every member of α. In case the members of α have no common part, there is no nucleus of α.

56.8. The negation $-x$ of an individual x is the fusion of all individuals discrete from x. The sum $x+y$ of individuals x and y is the fusion of the class having x and y as its only members. The product xy of individuals x and y is the nucleus of the class having x and y as its only members. There is no such product if x and y have no part in common.

CHAPTER 6

CALCULUS OF RELATIONS

60. Properties of relations analogous to properties of classes

60.1. INTRODUCTION. Two-termed relations may be viewed as being a special kind of propositional function of two arguments (05.9), just as classes may be viewed as being a special kind of propositional function of one argument. A proposition asserting a relation to hold between two individuals may be alternatively considered as a proposition asserting that an ordered couple of individuals belongs to a class. In other words, a (two-termed) relation may be viewed as a class of ordered couples. Note that in this chapter the term *relation* is reserved for two-termed relations, except in 67. In the first edition of [PM] the theory of relations (*21) is developed in close analogy with the theory of classes (50.4). Relations are treated there as propositional functions rather than as classes of ordered couples, and ordered couples themselves are viewed as being relations of a special kind called *ordinal couples* (*55). (For methods of representing relations by means of pair-lists, arrow-figures, and Schröder matrices, see, for example, Carnap 3523 (27, 28).)

60.2. VARIABLES, ABSTRACTS, AND CONSTANTS FOR RELATIONS

60.21. Systems dealing with relations usually employ special variables that range over relations. In [PM], for example, 'R', 'S', 'P', and 'Q' are used as such variables, though there seems to be a confusion there between such variables and variables that serve metalinguistic purposes (08.2–08.8).

60.22. Names of (two-termed) relations may be formed, in many systems, by placing two capped variables '$\hat{x}$' and '$\hat{y}$' to the left of a form '$(\ldots x\ldots y\ldots)$', giving an expression '$\hat{x}\hat{y}(\ldots x\ldots y\ldots)$' called a (*double*) *abstract* (40.7, 51.2). This notation appears in [PM], [R], and [Q]. Ways of defining such abstracts are given in [Q] 202 and in [R] 287. In [F] 167 the notation '$(xy\backslash(\ldots x\ldots y\ldots))$' is used instead of '$\hat{x}\hat{y}(\ldots x\ldots y\ldots)$'.

60.23. The *universal* relation (among individuals) is the relation that relates every individual to every individual. It may be denoted by '$\dot{V}$', where '$\dot{V}$' may be defined as '$\hat{x}\hat{y}[[x=x].[y=y]]$', letting '$x$' and '$y$' be variables for individuals ([PM] *25). If we wish to indicate type by a subscript, the notation '$V_{\alpha\iota}$' could be used in place of '$\dot{V}$' (32). The notation '$\dot{V}$' can also be used for the universal relation of any higher type. For example, the universal relation among things of type a could be expressed by '$\dot{V}$' or '$\hat{x}\hat{y}[[x=x].[y=y]]$' where '$x$' and '$y$' are variables of type a. If type is to be indicated explicitly, we would use '$V_{\alpha\alpha\alpha}$' in place of '$\dot{V}$'. An alternative way of defining '$\dot{V}$' is as '$\hat{x}\hat{y}\,\Upsilon$' (cp. 12.7). If no theory of types is used, it is possible to suppose that there is a universal relation that relates all entities whatsoever to one another, and not just the entities that belong to a specified type; in [F] 181 this relation is expressed by 'U^2' or '$U \times U$'.

60.24. The *empty* relation (among individuals) is the relation that relates no individuals. It may be denoted by '$\dot{\Lambda}$', where '$\dot{\Lambda}$' may be defined as '$\hat{x}\hat{y}[[x \neq x].[y \neq y]]$', letting '$x$' and '$y$' be variables for individuals ([PM] *25). If we wish to indicate type by a subscript (32), the notation '$\Lambda_{\alpha\iota}$' could be used in place of '$\dot{\Lambda}$'. The notation '$\dot{\Lambda}$' can also be used for the empty relation of any higher type. For example, the empty relation among things of type a could be expressed by '$\dot{\Lambda}$' or by '$\hat{x}\hat{y}[[x \neq x].[y \neq y]]$' where '$x$' and '$y$' are variables of type a. If type is to be indicated explicitly we would use '$\Lambda_{\alpha\alpha\alpha}$' in place of '$\dot{\Lambda}$'. An alternative way of defining '$\dot{\Lambda}$' is as '$\hat{x}\hat{y}\,\Lambda$' (cp. 12.7). If no theory of types is used it is possible to suppose that there is an empty relation which relates no entities and which is the complement (60.52) of the universal relation referred to at the end of 60.23.

60.3. NOTATION FOR ORDERED COUPLES AND RELATIONS

60.31. The ordered couple consisting of a and b, with a as the first member and b as the second member, is expressed by '$a;b$' in [Q] 198, by '$\langle a, b\rangle$' in [R] 280, and by '(a, b)' in [F] 84. The comma is often omitted in [F]. These notations for ordered couples are used by the above authors

in handling relations as classes of such ordered couples. On the other hand, [PM] has no notation for ordered couples in this sense, since relations are not treated in [PM] as classes of ordered couples; but the so-called ordinal couples are closely analogous to ordered couples and are defined as relations of a special kind (64.24, [PM] *55). Also see 64.26.

60.32. According to the definition of [Q] 206 and the equivalent definition of [R] 287, '$\hat{x}\hat{y}(\ldots x \ldots y \ldots)$' designates the class of all ordered couples $\langle a, b\rangle$ such that $(\ldots a \ldots b \ldots)$, and this class of ordered couples is viewed as being the relation of a to b such that $(\ldots a \ldots b \ldots)$. A similar standpoint is taken in [F] 167, but the notation '$(xy \backslash (\ldots x \ldots y \ldots))$' replaces '$\hat{x}\hat{y}(\ldots x \ldots y \ldots)$'.

60.33. To state that x and y stand in the relation R, the notation of [PM] *21 is 'xRy'. This is also essentially the notation of [R] 286 and [F] 88; we shall follow this notation. [Q] 200 writes '$R(x, y)$'. In the case of those authors who treat relations as classes of ordered couples, there are as alternatives to 'xRy' notations involving ordered couples. Thus we have '$x; y \in R$' in [Q] 198 (but see D24 in [Q] 200), '$R(\langle x, y\rangle)$' in [R] 281, and '$(xy) \in R$' or '$R(xy)$' in [F] 90, 91. '$R(xy)$' is equivalent to '$[xRy]$' or to 'xRy' according to a general rule of [F] 87 according to which '$[bac]$' is the abbreviation for '$a(bc)$'.

60.34. In [PM] *25.03 the notation '$\dot{\exists}!R$' is used to express the non-emptiness of the relation R, that is, the fact that R relates one or more things. This notation for relations is the analogue of '$\exists!\alpha$' for classes (52.3).

60.4. Functions from relations to propositions

60.41. Corresponding to each function from classes to propositions referred to in 53 there is an analogous function from relations to propositions. The definitions are as follows:

60.42. '$R \subset S$' may be regarded as equivalent to '$(x)(y)[xRy \supset xSy]$' (53.2) and as expressing that R is *relationally included* in S or that R is a *subrelation of* S. (See also 66.05.)

60.43. '$R \subseteq S$' may be written instead of '$R \subset S$', reserving '$\subset$' for the relation *proper subrelation* of as among relations (cp. 53.3) But this usage is not followed in this Dictionary nor in [PM]

60.44. '$R = S$' may be regarded as equivalent to '$(x)(y)[xRy \equiv xSy]$'. (Cp. 53.4.)

60.45. '$R \neq S$' may be regarded as equivalent to '$\sim[R = S]$'. (Cp. 53.5.)

60.46. [PM] *23.01 uses '$\in$' instead of '$\subset$' (60.42). Similarly [F] 185, 189 uses '$\subseteq_r$' and '$\simeq_r$' respectively in place of '$\subseteq$' and '$\simeq$' (53.3, 53.4) when dealing with relations rather than classes.

60.5. Functions from relations to relations

60.51. Corresponding to each function from classes to classes referred to in 54 there is an analogous function from relations to relations. Some of the more familiar ones are as follows:

60.52. The relation of x to y such that x does not bear R to y is called the (*relational*) *complement* of the relation R and is expressed by '$-R$'. We may regard '$-R$' as equivalent to '$\hat{x}\hat{y}(\sim[xRy])$'. Sometimes '$\bar{R}$' is written for '$-R$'. (Cp. 54.2.)

60.53. The relation of x to y such that x bears R to y and x bears S to y is called the (*relational*) *intersection* or *logical product* of the relation R with the relation S and is expressed by '$R \cap S$'. We may regard '$R \cap S$' as equivalent to '$\hat{x}\hat{y}[xRy \,.\, xSy]$'. (Cp. 54.3.) '$R - S$' may serve as an abbreviation for '$R \cap -S$'. (Cp. 54.51.)

60.54. The relation of x to y such that x bears R to y or x bears S to y is called the (*relational*) *union* or *join* or *logical sum* of the relation R with the relation S and is expressed by '$R \cup S$'. We may regard '$R \cup S$' as equivalent to '$\hat{x}\hat{y}[xRy \vee xSy]$'. (Cp. 54.4.)

60.55. [PM] (*23) writes '$\dot{-}$', '$\dot{\cap}$' and '$\dot{\cup}$' respectively for '$-$', '$\cap$' and '$\cup$'.

60.56. Other functions which are from relations to relations, but which have no analogues in the class calculus, are given subsequently in 61.

60.6. FUNCTIONS FROM CLASSES OF RELATIONS TO RELATIONS

60.61. If α is a class of relations, then the *relational intersection* (or *relational logical product*) of the class α of relations is defined as the relation that relates each pair that is related by every member of α (cp. 55.1). The relational intersection of the class α of relations may be denoted by $`\bigcap\alpha'$ which may be regarded as equivalent to $`\hat{x}\hat{y}(S)[S\in\alpha \supset xSy]'$.

60.62. If α is a class of relations, then the *relational union* (or *relational join* or *relational logical sum*) of the class α of relations is defined as the relation that relates each pair that is related by at least one member of α (cp. 55.2). The relational union of the class α of relations may be denoted by $`\bigcup\alpha'$ which may be regarded as equivalent to $`\hat{x}\hat{y}(\exists S)[S\in\alpha . xSy]'$.

60.63. The relation $\hat{p}$, *relational intersection of*, may be defined as $\hat{R}\hat{\alpha}[R = \hat{x}\hat{y}(S)[S\in\alpha \supset xSy]]$, and the relational intersection of α, $\bigcap\alpha$, may then be viewed as the relation that bears the relation $\hat{p}$ to α, that is, as $\hat{p}`\alpha$ (where the notation is that of 64.43 and of [PM] *30). Similarly $\hat{s}$, *relational union of*, may be defined as $\hat{R}\hat{\alpha}[R = \hat{x}\hat{y}(\exists S)[S\in\alpha . xSy]]$, and $\bigcup\alpha$ may be viewed as the relation that bears $\hat{s}$ to α, that is, as $\hat{s}`\alpha$ (cp. 55.4). (The letters '$\hat{p}$' and '$\hat{s}$' are not serving as metavariables in this paragraph but are the actual notation used in [PM].)

61. Relation-to-relation functions that are proper to the relational calculus

61.1. There are some functions from relations to relations which have no analogues among functions from classes to classes. The definitions of some of these are as follows:

61.2. $`\breve{R}'$ may be defined as $`\hat{y}\hat{x}[xRy]'$ ([PM] *31). Sometimes the notation $`\breve{}R'$ is used instead of $`\breve{R}'$. The relation $\breve{R}$ is called the *converse* of R. The relation $\breve{R}$ relates y to x if and only if the relation R relates x to y. A relation R is symmetrical if $\breve{R}\subset R$. A relation R is asymmetrical if $\breve{R}\subset -R$. The relation *similar to* is an example of a symmetrical relation; the relation *greater than* is an example of an asymmetrical relation. (See 65.1.)

61.3. $`R|S'$ may be defined as $`\hat{x}\hat{y}(\exists z)[xRz . zSy]'$ ([PM] *34). The relation $R|S$ is called the *relative product* of R and S. Let R be *debtor of* and S be *child of*. Then x has the relation $R|S$ to y if x is a debtor of a child of y, i.e., if there is at least one z such that x is a debtor of z and z is a child of y. The relative product satisfies the associative law, so that $[P|Q]|R = P|[Q|R]$. Consequently, $`P|Q|R'$ is often used as an abbreviation for $`[P|Q]|R'$ (16.5, [PM] *34.22). Schröder (III 29) used ';' instead of '|'.

61.4. $`R\dagger S'$ may be defined as $`\hat{x}\hat{y}(z)[xRz \vee zSy]'$ (Peirce 454*1*). The relation $R\dagger S$ is called the *relative sum* of R and S. Since the relation $\hat{x}\hat{y}[\sim[zSy] \supset [xRz]]$ can be shown to be equivalent to $R\dagger S$, the relation $R\dagger S$ relates x to y provided that x has the relation R to every individual not having the relation S to y. Similarly, the relation $R\dagger\breve{S}$ relates x to y provided that x has the relation R to every individual that has the relation S to y. Schröder (III 29) uses '$\dagger$' in place of '$\dagger$'.

61.5. Particular (relative) powers of a relation R may be defined thus:

$`R^{1}'$ may be defined as $`R'$,
$`R^{2}'$ may be defined as $`R|R'$ ([PM] *34.02),
$`R^{3}'$ may be defined as $`R^{2}|R'$ ([PM] *34.03),

and so on, for successively higher integral (relative) powers of R. (For a general definition of (*relative*) *powers of* R see [PM] *301.) If R is *parent of*, R^2 is *grandparent of*, R^3 is *great-grandparent of*, and so on. Sometimes the *zeroth power* is included among the powers of R, and $`R^{0}'$ may be defined as $`I \restriction C`R'$ (25.1, 62.4, 63.8, 64.13, 66.12), so that R^0 is the relation of identity as among things related by R. But in [Q] 253 the relation R^0 is defined as being the identity relation, I, itself. (See 08.9 with respect to 'I' and 'C'.)

61.6. $`R^{-1}'$, $`R^{-2}'$, $`R^{-3}'$, ..., are sometimes ([F] 182) written for $`\breve{R}'$, $`(\breve{R})^{2}'$, $`(\breve{R})^{3}'$, and so on. In addition to integral powers of R and negative integral powers of R, rational and real powers of R have sometimes been defined. (See *Journal of Symbolic Logic*, Vol. 14 (1949), pp. 81–84.)

61.7. A relation R is transitive if $R^2\subset R$. More intuitively, a relation R is said to be transitive provided that R holds between a and c whenever it holds between a and b and between b and c. For example, the relation

earlier than is transitive. The class of transitive relations is sometimes denoted by 'trans' ([PM] ∗201). A relation R is compact if $R \subset R^2$. Again, more intuitively, a relation R is said to be compact if whenever R holds between a and b, there is some c such that R holds between a and c and R holds between c and b. The relation *greater than* as among the rational numbers is compact.

62. Functions from classes to relations, or from classes and relations to relations

62.1. $`\alpha \uparrow \beta`$ may be defined as $`\hat{x}\hat{y}[x \in \alpha . y \in \beta]`$ ([PM] ∗35.04). $\alpha \uparrow \beta$ is the relation that relates each member of α to each member of β. If α is the class of males and β the class of females, then $\alpha \uparrow \beta$ is the relation which each male bears to each female. The relation $\alpha \uparrow \beta$ may be viewed as the universal relation with its domain restricted to α and its converse domain restricted to β. (For the concept of the universal relation, see 60.23. For the concepts of domain and converse domain, see 63.6, 63.7.)

62.2. $`\alpha \times \beta`$ is used instead of $`\alpha \uparrow \beta`$ in [R] 283 and [F] 180. This way of defining $`\alpha \times \beta`$ is consistent with 54.6, if relations are treated as being classes of ordered couples (60.1), because then $`\alpha \uparrow \beta`$ would denote the Cartesian product (in the sense of 54.6) of α with β. In [F] 180 the expression $`[\alpha^2]`$ is an abbreviation for $`\alpha \times \alpha`$, and $`[\alpha^3]`$ is an abbreviation for $`[\alpha^2] \times \alpha`$, and so on. We may view $[\alpha^2]$ as the universal relation with its field restricted to α. (For the concept of field, see 63.8.)

62.3. $`\alpha \restriction R`$ may be defined as $`\hat{x}\hat{y}[xRy . x \in \alpha]`$ ([PM] ∗35.01). We may speak of $\alpha \restriction R$ as the relation R with its domain restricted to α. As an example, let R be the relation *parent of*, α the class of *males*, and β the class of *females*. Then $\alpha \restriction R$ is the relation *father of*, and $\beta \restriction R$ is the relation *mother of*.

62.4. $`R \upharpoonright \beta`$ may be defined as $`\hat{x}\hat{y}[xRy . y \in \beta]`$ ([PM] ∗35.02). We may speak of $R \upharpoonright \beta$ as the relation R with its converse domain restricted to β. According to the example of 62.3, $R \upharpoonright \beta$ is the relation *parent to daughter*.

62.5. $`\alpha \restriction R \upharpoonright \beta`$ may be defined as $`\hat{x}\hat{y}[xRy . x \in \alpha . y \in \beta]`$ ([PM] ∗35.03).

We may speak of $\alpha \restriction R \upharpoonright \beta$ as the relation R with its domain restricted to α and its converse domain restricted to β. According to the example of 62.3, $\alpha \restriction R \upharpoonright \beta$ is the relation *father to daughter*.

62.6. $`R \restriction \alpha`$ may be defined as $`\hat{x}\hat{y}[xRy . x \in \alpha . y \in \alpha]`$ ([PM] ∗36.01). We may speak of $R \restriction \alpha$ as the relation R with its field restricted to α. According to the example of 62.3, $R \restriction \alpha$ is the relation *father to son*.

62.7. The following are definitions equivalent to those given in paragraphs 62.3 to 62.6:

62.71. $`\alpha \restriction R`$ may be defined as $`R \cap [\alpha \uparrow V]`$ or as $`R \cap [\alpha \times V]`$.

62.72. $`R \upharpoonright \beta`$ may be defined as $`R \cap [V \uparrow \beta]`$ or as $`R \cap [V \times \beta]`$.

62.73. $`\alpha \restriction R \upharpoonright \beta`$ may be defined as $`R \cap [\alpha \uparrow \beta]`$ or as $`R \cap [\alpha \times \beta]`$.

62.74. $`R \restriction \alpha`$ may be defined as $`R \cap [\alpha \uparrow \alpha]`$ or as $`R \cap [\alpha^2]`$.

63. Functions from relations to classes

63.1. The functions considered here are functions to classes from relations, or from relations and individuals, or from relations and classes. The notation of [PM] for such functions usually combines notation for relations of higher type, such as the notation $`\breve{R}`$ (66.08, [PM] ∗32.13), with the apostrophe used in connection with descriptive functions (64.42, [PM] ∗30). Thus the notation $`\breve{R}`$`'y` in [PM] is used to denote the class of things bearing R to y. The detailed discussion of these higher-type relations will be found in section 66.

63.2. $`R``\alpha`$ may be defined as $`\hat{x}(\exists y)[xRy . y \in \alpha]`$ ([PM] ∗37.01). $R``\alpha$ is the class of those entities which bear the relation R to members of α. In [Q] 209, $R``\alpha$ is referred to as the *map* or *projection* of $\alpha$ by $R$. [PM] calls $R``\alpha$ a *plural descriptive function* to oppose it to propositional functions while still indicating some analogy with so-called (singular) descriptive functions ([PM] ∗30; see 64.43). (More accurately, perhaps, we should speak of plural functional descriptions rather than plural descriptive

functions.) As an example, let R be the relation *parent of*. Then R"α is the class of parents of members of α.

63.3. It can be shown that $\breve{R}$"α (63.2) is the same as the class $\hat{y}(\exists x)[xRy . x \in \alpha]$ ([Q] 212). This class may be called the *retrojection* of α by R ([Q] 212). According to the example of 63.2, $\breve{R}$"α is the class of children of members of α.

63.4. Since $\{y\}$ is the unit class whose only member is y (64.2), R"$\{y\}$ (63.2) is the class of *R-predecessors* of y, that is, the class of things that bear R to y. Other notations equivalent to 'R"$\{y\}$' are '$\overrightarrow{R}$'y' (64.42, 66.08, [PM] *32.13), '$\hat{x}\{xRy\}$' (51.2), 'R"ιy' ([Q] 212), and '$R > y$' (Feys XV 71). According to the example of 63.2, R"$\{y\}$ is the class of parents of y.

63.5. Since $\{x\}$ is the unit class whose only member is x (64.2), $\breve{R}$"$\{x\}$ is the class of *R-successors* of x, that is, the class of entities which are borne R by x. Other notations equivalent to '$\breve{R}$"$\{x\}$' are '$\overleftarrow{R}$'x' (66.09, [PM] *32.131), '$\breve{R}$"ιx' ([Q] 213), '$\hat{y}\{xRy\}$' (51.2), and '$R < x$' (Feys XV 71). (In the combinatory notation of 41, an equivalent notation would be simply 'Rx'.) According to the example of 63.2, $\breve{R}$"$\{x\}$ is the class of children of x.

63.6. The class R"V is the same as the class $\hat{x}(\exists y)[xRy]$ ([Q] 212). R"V is the class of all R-predecessors, that is, the class of all entities that bear R to something or other. R"V is called the *domain* of R. Other notations equivalent to 'R"V' are '(Dm R)' ([F] 179) and '$\mathrm{D}'R$' ([PM] *33.4 ff.; see 66.11). According to the example of 63.2, R"V is the class of parents.

63.7. The class $\breve{R}$"V is the same as the class $\hat{y}(\exists x)[xRy]$ ([Q] 213). $\breve{R}$"V is the class of all the R-successors, that is, the class of all entities that are borne R by something or other. $\breve{R}$"V is called the *codomain* of R or the *converse domain* of R. Other notations equivalent to '$\breve{R}$"V' are '(Cndm R)' ([F] 179) and '$\mathrm{C}'R$' ([PM] *33.111; see 66.11). According to the example of 63.2, $\breve{R}$"V is the class of children.

63.8. The class $[R \cup \breve{R}]$"V is the same class as $\hat{x}(\exists y)[xRy \vee yRx]$. This class is called the *field* of R; it is the class of all entities which bear the relation R to something or other or which are borne the relation R by something or other. Other notations equivalent to '$[R \cup \breve{R}]$"V' are '(Fld R)' ([F] 179) and '$\mathrm{C}'R$' ([PM] *33.112; see 66.12) According to the example of 63.2, $[R \cup \breve{R}]$"V is the class of all parents and all children. (See 08.9 with respect to 'C'.)

63.9. 'R"α' may be defined as '$\hat{x}(y)[y \in \alpha \supset xRy]$'. This notation, suggested in Feys XV 71, denotes the class of individuals which have the relation R to every member of α. A notation equivalent to 'R"α' is '$p'\overrightarrow{R}$"α' ([PM] *40.51; see 66.02). If R is the relation *less than* among numbers, R"α is the class of numbers which are less than every member of α. (See 08.9 with respect to 'p'.)

64. Identity and notions derived from identity

64.1. Identity

64.11. The symbol 'I', already introduced in 25.1, is often used to denote the relation of *identity* ([PM] *50.01; [Q] 229; [R] 322). For the distinction in [Q] between '$=$' and 'I' see [Q] 230 In combinatory logic the symbol 'Q' is often used to denote identity, as in 42.2 and in [CF] 241. (The use of 'I' to denote identity should not be confused with the use of the combinator 'I' in combinatory logic (45.1). Similarly for 'J' below.)

64.12. The symbol 'J' is often used to denote the relation of *non-identity* (sometimes called *diversity*). We may treat 'J' as an abbreviation for '$-I$' ([PM] *50.02).

64.13. 'R^0' may be defined as '$I \restriction [[R \cup \breve{R}]$"V]' (Carnap 3523, 38). R^0 is identity within the field of R (63.8); it may be considered as the *zeroth power* of R. If we are reasoning, for example, about the relation R of being alike in color, it is often useful to consider identity only as among things having color (that is, only as among members of the field of R) rather than considering the relation of identity also as among things not having color. (On the other hand, in [Q] 253, the relation of identity itself, I, is treated as being R^0 (cp. 61.5).) (See 08.9 with respect to 'I'.)

64.14. A relation R is *reflexive* if $R^0 \subset R$ (Carnap 3523, 40), that is, if it relates each thing to itself that is in the field of R (63.8). A relation R is *totally reflexive* if $I \subset R$ (Carnap 3523, 40). The relation of being alike in color is a reflexive relation, but not a totally reflexive one. The relation of self-identity is totally reflexive. The relation of being alike in color is not totally reflexive since there are many things, such as numbers, which have no color and hence no likeness in color, even to themselves.

64.15. A relation R is *connected* if $[J \upharpoonright [[R \cup \breve{R}]^{\prime\prime} V]] \subset [R \cup \breve{R}]$ (Carnap 3523, 65). That is, a relation R is connected if for every pair x, y in the field of R such that x is not identical with y, either x bears R to y or y bears R to x (or both). The expression 'connex' is often used to denote the class of connected relations ([PM] *202.01). Precedence among numbers is a connected relation since if x and y are two different numbers (two different members of the field of the relation in question), then either x precedes y or y precedes x.

64.2. Unit classes

64.21. '$\{y\}$' may be defined as '$\hat{x}[xIy]$' or equivalently as '$\hat{x}[x=y]$' ([F] 104). $\{y\}$ is the class of entities identical to y. This class has y as its only member and so is called a *unit class*. Ordinarily y and $\{y\}$ must not be identified: if y is Rubens, $\{y\}$ is not Rubens but is rather the class having Rubens as its only member. (In [Q] 135 the class $\{y\}$ is identified with y itself if y is an individual in Quine's sense.) Other notations equivalent to '$\{y\}$' are '$\iota'y$' ([PM] *51.01; see 64.42), '$\imath y$' ([Q] 189), '$I \rangle y$' (according to 63.4), and 'Qy' (41.4). (See 08.9 with respect to 'I'.)

64.22. '$\{x, y\}$' may be defined as '$\{x\} \cup \{y\}$' ([R] 249). Thus $\{x, y\}$ is the class which is the union (54.4) of the unit classes $\{x\}$ and $\{y\}$, so it is the class which has x and y as its only members. Notice that $\{x, x\}$ is the same class as $\{x\}$. Sometimes $\{x, y\}$ is called the *unordered couple* ([F] 105) or the *cardinal couple* ([PM] *54) of x and y. An alternative notation is '$\iota'x \cup \iota'y$' ([PM] *54).

64.23. Similarly, '$\{x, y, z\}$' may be defined as '$\{x\} \cup \{y\} \cup \{z\}$', and '$\{x, y, z, w\}$' may be defined as '$\{x\} \cup \{y\} \cup \{z\} \cup \{w\}$', and so on. Thus '$\{x, y, z\}$' is the class having x, y and z as its only members.

64.24. '$z \downarrow w$' may be defined as '$\{z\} \uparrow \{w\}$' (62.1, [PM] *55.01). Thus $z \downarrow w$ is the relation that holds only from z to w. This relation $z \downarrow w$ is called the *ordinal couple* of z with w. Notice that $z \downarrow w$ is a relation, whereas the couples of 60.31, written there as '$w;z$' or '$\langle w, z \rangle$' or '(w, z)' are not relations but may be viewed as members of relations if relations are treated as classes of ordered couples (60.1). (See also 64.26.)

64.25. For the notations '$R^{\prime\prime}\{y\}$' and '$\breve{R}^{\prime\prime}\{x\}$' see 63.4 and 63.5.

64.26. The notation for *pair* or *ordered couple* '$\langle a, b \rangle$' is defined by Wiener (2381) as '$\{\{a\}, \{a, b\}\}$'. Thus an ordered couple may be defined as a class whose only members are the unit class of a and the unordered couple $\{a, b\}$. (See also 60.3.)

64.3. One-many, many-one, and one-one relations

64.31. A relation R is *one-many* if it never relates two different things to some one thing. The class of one-many relations is symbolized by '1→Cls' in [PM] *71, and '1→Cls' may be regarded as an abbreviation for '$\hat{R}(x)(y)(z)[[xRz \,.\, yRz] \supset x=y]$' in virtue of [PM] *71.17. (Actually, in [PM] *70 a more general notion represented by '$\alpha \rightarrow \beta$' is first defined, and then the concepts represented by '1→Cls', 'Cls→1', and '1→1' are treated as special cases of the general concept, where, in accordance with [PM] *20.03, 'Cls' stands for the class of all classes of some fixed type.)

64.32. A relation R is *many-one* if it never relates any one thing to two different things. The class of many-one relations is symbolized by 'Cls→1' in [PM] *71, and 'Cls→1' may be regarded as an abbreviation for '$\hat{R}(x)(y)(z)[[xRy \,.\, xRz] \supset y=z]$' in virtue of [PM] *71.171. A relation is many-one (one-many) if and only if its converse is one-many (many-one) ([PM] *71.21).

64.33. A relation R is *one-one* if it is both one-many and many-one. The class of one-one relations is symbolized by '1→1' in [PM] *71, and '1→1' may be regarded as an abbreviation for '[1→Cls]∩[Cls→1]' in virtue of [PM] *71.04. A relation is one-one if both it and its converse are one-many.

64.4. Relations associated with functions

64.41. Every function f gives rise to a one-many relation R which relates

each value of f to the argument or arguments for which f has that value. Let us say that R is the one-many relation *associated with* the function f. Given any statement about the function f, there will be some equivalent statement about the associated one-many relation R, and vice versa.

64.42. In some systems of logic, such as [PM], the only functions considered are propositional functions, and so statements apparently about non-propositional functions are viewed as statements about the one-many relations associated with these functions (since relations, or relationships, can themselves be regarded as propositional functions (05.9)). Thus if f is a non-propositional function having the associated one-many relation R, we write '$R`a$' in [PM] in place of '$f(a)$' and we treat '$R`a$' as an abbreviation for '$(\iota x)[xRa]$' (see 26 and [PM] *30.01). Thus '$R`a$' denotes the thing which bears R to a, and hence denotes the same thing as '$f(a)$'. This device for representing non-propositional functions by their associated relations is feasible only when the associated relations are definable in the system of logic in question. In [PM] the required relations are always definable.

64.43. Expressions of the form '$R`a$' are sometimes called *descriptive functions* ([PM] *30); but since these expressions are names of values of functions (or at least act as variable expressions if 'a' is a variable) it would seem better to call these expressions *functional descriptions* and view them as a special kind of definite description (26), which, indeed, they are.

64.44. All considerations applying to definite descriptions of course apply also to functional descriptions, including matters concerned with scope. In the system of [PM] if '$\mathrm{E}!R`x$' can be proved, then the extent of the scope of '$R`x$' makes no difference. (See 26.5 and [PM] *30.) Here '$\mathrm{E}!R`x$' means that the "$R$ of $x$" exists uniquely, that is, that there is one and only one thing which bears $R$ to $x$. If '$\mathrm{E}!R`x$' can be proved in [PM], then '$R`x$' acts like a name in [PM]. The following statement is logically equivalent to the assertion that $R$ is one-many: $(x)[xRy \supset \mathrm{E}!R`y]$.

64.45. In [PM] *37.04 the notation '$\mathrm{E}!!R``\alpha$' is used as an abbreviation for '$(x)[x \in \alpha \supset \mathrm{E}!R`x]$'. We can read '$\mathrm{E}!!R``\alpha$' as asserting that the R of each member of α exists uniquely or that there is one and only one thing, for each member of α, that bears R to that member of α.

64.46. The relation ι may be defined as the relation that relates each unit class to the only member of that unit class, thus 'ι' may be treated as abbreviating '$\hat{\alpha}\hat{x}[\alpha = \{x\}]$'. This relation is especially relevant to the theory of definite descriptions, and the functional description, '$\iota`a$', may be used as an alternative notation to '$\{a\}$' to signify the class whose only member is $a$ (64.2). The converse of $\iota$ may be written as '$\breve{\iota}$' or as '$\imath$', and the functional description, '$\imath`a$', denotes the only member of α if α is a unit class. Similarly '$\imath`\hat{x}(\ldots x\ldots)$' denotes the only thing in the class $\hat{x}(\ldots x\ldots)$, that is, the thing x that satisfies the condition $(\ldots x\ldots)$ (provided that there is only one such thing). It is therefore clear that '$(\iota x)(\ldots x\ldots)$' has essentially the same interpretation as '$\imath`\hat{x}(\ldots x\ldots)$', and this indicates why the inverted iota is used in definite descriptions such as '$(\iota x)(\ldots x\ldots)$'. In [PM] *51.01 the symbol 'ι' is defined as '$\vec{I}$'. (See 63.1, 64.1, and 66.08.) In [PM] the converse of ι is denoted by '$\breve{\iota}$' rather than by '$\imath$'.

65. Representation of non-propositional functions by their associated relations

65.1. Many systems of logic, including [PM], can deal formally only with propositional functions and not with other kinds of functions, that is, not with non-propositional functions. Fortunately, however, these other functions (for example, functions from classes to classes) can be adequately represented by their associated relations (64.42). For example, the notion of *converse* of a relation (61.2) is a function from relations to relations. In [PM] we define the associated relation Cnv as $\hat{R}\hat{S}[R = \breve{S}]$, and let this relation represent the function *converse of*. The functional description '$\mathrm{Cnv}`S$' would then denote the value of this function when 'S' denotes its argument. The function itself is not actually denoted by any expression, but the corresponding associated relation is denoted by 'Cnv'. Similarly, the function *domain of* is not denoted by any expression, but the corresponding associated relation $\hat{\alpha}\hat{R}[\alpha = \hat{x}(\exists y)[xRy]]$ is denoted in [PM] by the symbol 'D'. Hence '$\mathrm{D}`R$' may be read as *the domain of* R.

65.2. In general, if there is a non-propositional function such that

'$(\ldots a\ldots)$' denotes the value of the function if 'a' denotes its argument, then the associated relation can be denoted by '$\hat{x}\hat{y}[x = (\ldots y\ldots)]$', and the latter expression may be abbreviated as '$\lambda_y(\ldots y\ldots)$', as in [Q] 226. Whereas in lambda-conversion '$\lambda y(\ldots y\ldots)$' would denote the function itself (44), in systems like [Q], which lack a direct notation for non-propositional functions, the notation '$\lambda_y(\ldots y\ldots)$' may be used for denoting the associated relation. Here '$(\lambda_y(\ldots y\ldots))`a$' plays the role played by '$(\lambda y(\ldots y\ldots))\,a$' in lambda-conversion, and we have $(\lambda_y(\ldots y\ldots))`a = (\ldots a\ldots)$, just as we have $(\lambda y(\ldots y\ldots))a$ cnv $(\ldots a\ldots)$ in lambda-conversion. Also, we always have '$\mathrm{E!}\lambda_y(\ldots y\ldots)`a$' as a theorem, since the associated relation is one-many and non-empty.

65.3. The method described above for representing non-propositional functions by their associated relations is available in systems like [PM] and [Q] only if the arguments and values of the non-propositional functions are themselves propositional functions of individuals; but this limitation produces no difficulty in practice, because we can always find corresponding non-propositional functions whose values and arguments, if not individuals, are themselves associated relations (of non-propositional functions) and hence themselves propositional rather than non-propositional functions.

65.4. In [PM] (*38) abbreviations of the following sort are introduced

'$\alpha\cap$' for '$\hat{\gamma}\hat{\beta}[\gamma = [\alpha\cap\beta]]$',
'$R\mid$' for '$\hat{P}\hat{S}[P = [R\mid S]]$',
'$R\restriction$' for '$\hat{S}\hat{\alpha}[S = [R\restriction\alpha]]$',

and so on. Also

'$\cap\beta$' for '$\hat{\alpha}\hat{\gamma}[\gamma = [\alpha\cap\beta]]$',
'$\mid R$' for '$\hat{P}\hat{R}[P = [R\mid S]]$',
'$\restriction\alpha$' for '$\hat{S}\hat{R}[S = [R\restriction\alpha]]$',

and so on. In general, if '0' is a symbol that can be thought of as representing a binary operation, then '$0b$' abbreviates '$\hat{x}\hat{y}[x = [y0b]]$' and '$a0$' abbreviates '$\hat{x}\hat{y}[x = [a0y]]$'. The advantage of such notation is that we then can prove both

$$a0`b = [a0b]$$

and

$$0b`a = [a0b]$$

where '$a0$' denotes the associated relation of the function that has the value $a0b$ for the argument b, and '$0b$' denotes the associated relation of the function that has the value $a0b$ for argument a. Sometimes to avoid ambiguity we use '$(a0)$' instead of '$a0$'.

65.5. Combining the methods of abbreviation referred to in 65.2 and 65.4, we could treat '$0b$' as an abbreviation for '$\lambda_y[y0b]$' and '$a0$' as an abbreviation for '$\lambda_y[a0y]$', but [PM] (*38) does not use this lambda-notation.

65.6. If R is the relation $a0$ of 65.4 and 65.5, then R relates $a0b$ to b for every b, so that $[a0b]Rb$ is true, that is, $[a0b]\,(a0)\,b$. Hence $a0``\beta$ would be the class of those things $a0y$ such that $y$ is a member of $\beta$. For example, $R\mid``\beta$ is the class of those relative products $R\mid S$ such that the relation S is a member of β, and $a\downarrow``\beta$ is the class of those ordinal couples $a\downarrow x$ such that $x$ is in $\beta$. Similarly $0b``\alpha$ is the class of those things $x0b$ such that x is a member of α. For example, $\mid S``\alpha$ is the class of those relative products $R\mid S$ such that R is a member of α.

65.7. It is sometimes desirable to view the class $0b``\alpha$ as if it were the result of an operation on $\alpha$ and $b$. Hence we treat '$\alpha\,\ddot{0}\,b$' as an abbreviation for '$0b``\alpha$' ([PM] *38.03) and regard '$\ddot{0}$' as standing for an operation on α and b. Then '$\alpha\,\ddot{0}``\beta$' would denote the class of those classes $0b_1``\alpha, 0b_2``\alpha$, and so on, where $b_1, b_2$, and so on are members of $\beta$, and where, for example, $0b_2``\alpha$ is the class of those things $a_1 0b_2, a_2 0b_2, a_3 0b_2$, and so on, where a_1, a_2, a_3, and so on are members of α. More specifically, if '$\ddot{0}$' is '$\ddot{\downarrow}$', where '$a\downarrow b$' denotes the ordinal couple a, b (64.24, [PM] *55), then '$\alpha\,\ddot{\downarrow}\,b$' would denote the class of all ordinal couples whose first element is in α and whose second element is b, and '$\alpha\,\ddot{\downarrow}``\beta$' would denote the class of all classes $\alpha\,\ddot{\downarrow}\,b$ of ordinal couples where $b$ is in $\beta$. Hence, '$s`[\alpha\,\ddot{\downarrow}``\beta]$' denotes the logical sum of the class of classes $[\alpha\,\ddot{\downarrow}``\beta]$ (55.4), and hence denotes the class of all those ordinal couples such that the first element is in $\alpha$ and the second element is in $\beta$. Finally it can be shown that $s`s`[\alpha\,\ddot{\downarrow}``\beta]$ (60.63) is the same as $\alpha\restriction\beta$ (62.1). In the notation of [R]222 (cp. 51.3) the

class s'$[\alpha \downarrow ``\beta]$ could be denoted by '$\{x \downarrow y | x \in \alpha, y \in \beta\}$'. This class is also the same as $\beta \times \alpha$ as defined in [PM] *113.02, where it is called the *arithmetic class-product of α and β*. It is closely analogous to the Cartesian product of α and β (54.6, 62.1, 62.2) but the latter is a class of ordered couples (60.3) rather than a class of ordinal couples (64.24). See also 60.1. (Cp. 08.9 with respect to 's'.)

66. Some further important relations and classes

66.01. Quine in 4585, p. 26, writes '$\langle \alpha, x \rangle$' for '$x \in \alpha$'. (Cp. 52.1)

66.02. The symbol '$\in$', though not originally denoting a relation in [PM] *20.20, is nevertheless also used in *62.01 as the name of the relation of membership. Thus $\in ``\kappa$ is the sum of the class $\kappa$ of classes (55.2, 55.4, 63.4, [PM] *62.3), and $\in ``\kappa$ is the product of the class κ of classes (55.1, 55.4, 63.9). '$\in$' is used to denote membership in [Q] (232). (In the foregoing paragraph and hereafter, the Greek letter 'κ' is used like italic letters for metalinguistic purposes. See 08.3–08.8 and especially 08.7.)

66.03. The symbols '$\notin$' and '$\ni$' may be used as abbreviations for '$-\in$', that is, as names for non-membership (cp. [R] 197).

66.04. Quine in 4585, p. 78, writes '$\exists x$' for '$\hat{\alpha}(\alpha, x)$'. Here '$\exists x$' denotes the class of classes of which x is a member.

66.05. 'Cl' is an abbreviation for '$\hat{\kappa}\hat{\alpha}[\kappa = \hat{\beta}[\beta \subset \alpha]]$' in [PM] *60.01. Thus the relation Cl holds from a class κ (of classes) to a class α if κ is the class of subclasses of α. The class Cl'α is the class of subclasses of α and is therefore the same as $\hat{\beta}[\beta \subset \alpha]$. In [R] (255) '$\mathrm{SC}(A)$' stands for '$\hat{\beta}(\beta \subset A)$'. Here Rosser uses '$\subseteq$' instead of '$\subset$', the symbol used in [PM], and Rosser would reserve '$\subset$' for proper inclusion (see 53.3). In [PM] '$\subset$' does not mean proper inclusion. In [R] (252) '$\mathrm{USC}(A)$' stands for '$\{\{x\} | x \in A\}$' and denotes the class of unit subclasses of A (cp. 51.3, 64.2). Just as 'Cl' denotes the relation of the class of subclasses of a class to the class itself, so also 'Rl' is used to denote the relation of the class of subrelations (60.42) of a given relation to that relation itself. Accordingly, 'Rl' is an abbreviation for '$\hat{R}\hat{R}[\kappa = \hat{S}[S \in R]]$' (60.46, 60.42, [PM] *61.01).

66.06. In Quine 4585, p. 35, the expression '$[\alpha]$' stands for the same class that '$\hat{\beta}[\alpha \subset \beta]$' stands for in the usual notation, that is, it stands for the class of classes of which α is a subclass.

66.07. 'Cnv' for '$\hat{R}\hat{S}[R = \breve{S}]$' (here "for" means "is an abbreviation for" and, of course, 'R' and 'S' are variables for relations). '$\mathrm{Cnv}`R$' means the same as '$\breve{R}$'. (Cp. 61.2, [PM] *31.01.)

66.08. '$\vec{R}$' for '$\hat{\alpha}\hat{y}[\alpha = \hat{x}[xRy]]$'. '$\vec{R}`y$' means the same as '$\hat{x}[xRy]$'. (Cp. 63.2, [PM] *32.01.) If we call $\vec{R}`y$ the class of *R-predecessors* of y (the class of those things which bear R to y), then $\vec{R}``\beta$ can be referred to as the class of classes of R-predecessors of members of β. (Cp. 63.4)

66.09. '$\overleftarrow{R}$' for '$\hat{\beta}\hat{x}[\beta = \hat{y}[xRy]]$'. '$\overleftarrow{R}`x$' means the same as '$\hat{y}[xRy]$'. (Cp. 63.3, [PM] *32.02.) If we call $\overleftarrow{R}`x$ the class of *R-successors* of x (the class of those things to which x bears R), then $\overleftarrow{R}``\alpha$ can be referred to as the class of classes of R-successors of members of α. (Cp. 63.5.)

66.10. 'sg' for '$\hat{S}\hat{R}[S = \vec{R}]$', and 'gs' for '$\hat{S}\hat{R}[S = \overleftarrow{R}]$'. '$\mathrm{sg}`R$' means the same as '$\vec{R}$', and '$\mathrm{gs}`R$' means the same as '$\overleftarrow{R}$'. (Cp. 66.08, 66.09, [PM] *32.02, *32.04.)

66.11. 'D' for '$\hat{\alpha}\hat{R}[\alpha = R``\mathrm{V}]$', and '$\mathrm{C}$' for '$\hat{\alpha}\hat{R}[\alpha = \breve{R}``\mathrm{V}]$'. '$\mathrm{D}`R$' means the same as '$R``\mathrm{V}$', that is, the domain of $R$; and '$\mathrm{C}`R$' means the same as '$\breve{R}``\mathrm{V}$', that is, the codomain of R. (Cp. 63.6, 63.7, [PM] *33.)

66.12. 'C' for '$\hat{\alpha}\hat{R}[\alpha = R \cup \breve{R}]``\mathrm{V}$'. '$C`R$' means the same as '$[R \cup \breve{R}]``\mathrm{V}$', that is, the field of R. (Cp. 08.9, 63.8, [PM] *33.)

66.13. If f is a function and if R is the associated relation of f, then $\mathrm{D}`R$, the domain of $R$, is the class of values of $f$, while $\mathrm{C}`R$, the codomain of R, is the domain of definition (05.3) of f. Rosser, however, lets the *converse* of the associated relation serve, for his purposes, as the associated relation, and he treats the function as identical with its associated relation (in his sense). Thus instead of saying that the relation associated with a function f is the one-many relation, $\hat{x}\hat{y}[x = f(y)]$ (cp. 64.31), Rosser would say that the relation associated with a function f is the many-one relation,

$\hat{x}\hat{y}[f(x) = y]$ and that f itself is simply this many-one relation. This procedure is in close accordance with present-day usage in the field of mathematics, since the domain of the relation is usually treated as the domain of definition of the function (that is, the class of arguments of the function) and the codomain of the relation is usually treated as the class of values of the function. Thus Rosser defines '$\mathrm{Arg}(R)$' as '$\hat{x}(\exists y)[xRy]$' and '$\mathrm{Val}(R)$' as '$\hat{y}(\exists x)[xRy]$', and '$\mathrm{AV}(R)$' as '$\mathrm{Arg}(R) \cup \mathrm{Val}(R)$', so that '$\mathrm{Arg}(R)$' denotes the domain of the relation, and '$\mathrm{Val}(R)$' denotes the codomain of the relation, and '$\mathrm{AV}(R)$' denotes the union of $\mathrm{Arg}(R)$ and $\mathrm{Val}(R)$ and so denotes the field of R.

66.14. Quine, unlike Rosser, treats the associated relation of a function as one-many rather than many-one and identifies functions with their associated one-many relations. For any relation R, whether one-many or not (that is, whether a function or not, in Quine's terminology), there is a class $\imath R$ called by Quine the "range of functionality of R" and having as members those members of the codomain of R each of which is borne the relation R by exactly one member of the domain of R. Thus '$\imath R$' may be defined as '$\hat{y}(\exists z)(x)[x = z \equiv xRy]$' (cp. [Q] 222). A relation whose codomain is its range of functionality would then be one-many and hence a function in Quine's sense.

66.15. '$\breve{F}$' for '$\breve{\alpha}|C$', though a more complicated definition is given in [PM] *33.04. $\breve{F}$ is the relation of each member of the field of a relation to the relation. $\breve{F}$ is the same relation as C. (Cp. 08.9 with respect to 'C' and 'F'.)

66.16. 'B' for '$\hat{x}\hat{R}[x \in \mathrm{D}'R . \sim[x \in \breve{\mathrm{C}}'R]]$' (Woodger III 42, p. 40). The relation B (beginner of) relates x to R if x bears R to something but is not borne R by anything. (Cp. 08.9 with respect to 'B'.)

66.17. 'R_{α}' for '$\hat{\alpha}\hat{\beta}[\alpha = \breve{R}`\beta]$'. ([PM] *37.02.) '$R_{\alpha}$' denotes the same relation as '$\overrightarrow{R|\alpha}$' (61.3, 66.08). The relation R_{α} relates a class α to a class β provided that α consists of all those things which bear R to members of β. For example, if R is the relation *husband of*, then R_{α} is the relation of α to β such that α is the class of husbands of members of β. Thus, $R_{\alpha}`\beta$ is the same class as $\breve{R}`\beta$. By the convention stated in 65.4, it is possible

to regard 'R_{α}' as denoting a relation, in fact the same relation as is denoted by 'R_{α}'.

66.18. '$R``\kappa$' for '$R_{\alpha}``\kappa$', or for '$(\breve{R})``\kappa$'. ([PM] *37.04.) Thus '$R``\kappa$' denotes the class of all classes each of which bears the relation R_{α} to some member of κ. For example, if R is the relation *husband of*, if κ_1 consists of classes of men and κ_2 consists of classes of women, and if each class in κ_1 consists of the husbands of the members of some class in κ_2, then κ_1 is $R``\kappa_2$.

66.19. If 'R' denotes any relation, 'R_{*}' denotes the *ancestral of* R, that is, the relation that holds from x to y if some power (including possibly the zero power) of R holds from x to y (61.5). Thus a bears R_{*} to b provided that there exists a finite number n of steps such that $aRc_1, c_1Rc_2, \ldots, c_nRb$ are all true, or if (in the case for the zero power) a is identical with b and is in the field of R. The relation R_{*} is called the ancestral of R because if R is the relation *parent of* then R_{*} is the relation *ancestor of* assuming that we agree to count a person as his own ancestor. It can be shown that R_{*} is reflexive (64.14) and transitive (61.7). In order to explain the definition of 'R_{*}' as given in [PM] *90.01, it is desirable to introduce the concept of *R-hereditary class*. A class α will be said to be *R-hereditary* if everything that is borne R by a member of α is also a member of α. Thus α is R-hereditary if and only if $\breve{R}``\alpha \subset \alpha$. The relation R_{*} may then be defined as the relation of x to y such that x belongs to the field of R and y belongs to every R-hereditary class to which x belongs. For example, if R is the relation *father of* and if α is the class of people having the surname "Adams", than α is R-hereditary, because a child (in most English-speaking societies) has the same surname as its father. Clearly the only condition under which y can belong to *all* the R-hereditary classes to which x belongs is the condition that there exists a finite sequence of true propositions of the form: $xRz_1, z_1Rz_2, \ldots, z_{n-1}Rz_n, z_nRy$, and hence that x is an ancestor of y along the male line. In fact R_{*} in this example is the relation *ancestor of along the male line*. These considerations justify the definition of 'R_{*}' in [PM] *90.01 as '$\hat{x}\hat{y}[[x \in C`R] . (\alpha)[[[\breve{R}``\alpha \subset \alpha] . [x \in \alpha]] \supset [y \in \alpha]]]$'. For another slightly different form of this definition that shows the relation of the concept of the ancestral to the mathematical

concept of closure, see [R] 305. In [Q] 216, a slightly different kind of ancestral, $*R$, is defined. This has the property $(x)*R(x, x)$ and is consistent with Quine's view of R^0 as being identity. See 64.13 and [Q] 221, 253. (See 08.9 with respect to 'C'.)

66.20. Just as R_* is the relation of x to y such that some power of R, including possibly the zero power, holds from x to y, so R_{po} is the relation of x to y such that some power of R, other than the zero power, holds from x to y. We can regard 'R_{po}' as an abbreviation for '$R_*|R$' (cp. [PM] *91.154), though the definition actually used in [PM] *91.05 is '$\hat{S}$'Pot'R' (cp. 60.63 and 66.23), so that R_{po} is viewed as the relational sum of the class of non-zero powers of R. In [Q] 221 R_{po} is called the *proper ancestral*.

66.21. The relation $R_{\downarrow\downarrow}$ is the relation that each of the relations S, $R|S$, $R|(R|S)$, $R|(R\ (R|S))$, and so on bears to S. We can define '$R_{\downarrow\downarrow}$' as '$(R|)_*$' ([PM] *91.01), since $R|S$ bears $R|$ to S, and $R|(R|S)$ bears $R|$ to $R|S$, and so on, so that each of $R|S$, $R|(R|S)$, and so on, bears $(R|)_*$ to S (cp. 65.4, 61.3, 66.19), and S itself bears $(R|)_*$ to S because of the reflexivity of $(R|)_*$ (66.19).

66.22. Similarly the relation $R_{\uparrow\uparrow}$ is the relation that each of the relations S, $S|R$, $(S|R)|R$, and so on, bears to S, and '$R_{\uparrow\uparrow}$' may be defined as '$(|R)_*$' ([PM] *91.02). In fact, $R_{\uparrow\uparrow}$ relates each non-zero power of R (61.5) to R itself.

66.23. The relation Pot is the relation that the class of powers of R, exclusive of the zero power, bears to R (61.5). Hence 'Pot' may be defined as '$\hat{R}\hat{R}[\kappa = \hat{S}[SR_{\downarrow\downarrow}R]]$', and '$\mathrm{Pot}'R$' denotes the class of powers of R.

66.24. The relation Potid is the relation of the class of all powers of R, including the zero power, to R itself, so that $\mathrm{Potid}'R$ is the class of all powers of R, including the zero power. We may define 'Potid' as '$\hat{R}\hat{R}[\kappa = \hat{S}[SR_{\downarrow\downarrow}R^0]]$' where '$R^0$' is defined as '$I \restriction C'R$' (61.5). Here 'id' is used as part of 'Potid' to indicate that the zero power, which is a kind of restricted identity, is included among the powers. (See 08.9 with respect to 'C' and 'I'.)

66.25. The R-posterity of x consists of all things to which x bears R_*, and hence it is $\overleftarrow{R_*}'x$. The proper R-posterity of x consists of all things to which x bears R_{po} (the proper ancestral of R), and hence it is $\overleftarrow{R_{po}}'x$ (66.20). Similary, R-ancestry of x is $\overrightarrow{R_*}'x$ and the proper R-ancestry of x is $\overrightarrow{R_{po}}'x$. It is not always the case that x is in its own proper R-ancestry, though x is always in its own R-ancestry if x is in the field of R.

66.26. '$R(x-y)$' for '$\overleftarrow{R_{po}}'x \cap \overrightarrow{R_{po}}'y$' ([PM] *121.01). Here $R(x-y)$ is called the *interval* between x and y, not including the endpoints x and y. It consists of those things which both belong to the proper R-posterity of x and also belong to the proper R-ancestry of y. Similarly '$R(x-\!\!|\ y)$', '$R(x|\!\!-y)$', and '$R(x|\!\!-|\ y)$' are respectively defined as '$\overleftarrow{R_{po}}'x \cap \overrightarrow{R_*}'y$' '$\overleftarrow{R_*}'x \cap \overrightarrow{R_{po}}'y$', and '$\overleftarrow{R_*}'x \cap \overrightarrow{R_*}'y$'. In the interval $R(x-\!\!|\ y)$ the endpoint y is included, but not the endpoint x. In the interval $R(x|\!\!-y)$ the endpoint x is included, but not the endpoint y. In the interval $R(x|\!\!-|\ y)$ the endpoints x and y are both included.

66.27. '$R;S$' for '$R|S|\breve{R}$' ([PM] *150.01), or equivalently, for '$\hat{x}\hat{y}(z, w)$ $[xRz.zSw.yRw]$'. Thus $R;S$ holds from a to b if a bears R to z, and z bears S to w, and y bears R to w. If R is one-one, then $R;S$ is the relation of $R'z$ to $R'w$ provided that z bears S to w. For example, if S is some relation between numbers, and if R is the relation *square of*, then $R;S$ is the relation between the squares of two numbers when S is the relation between the numbers themselves. There is an analogy between $R;S$ and $R''\alpha$ (63.2) in the following sense: If we view R and S as classes of ordered couples and if P is the relation that relates the ordered couple $\langle x_1, x_2\rangle$ to the ordered couple $\langle y_1, y_2\rangle$ provided that $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ are both members of R (i.e., provided that R relates x_1 to y_1 and relates x_2 to y_2), then $R;S$ is the same relation (that is, the same class of ordered couples) as $P''S$. (Cp. 60.1, 60.3.) Just as $R''\alpha$ is called the *map* or *projection* of α by R, so also $R;S$ may be called the *map* or *projection* of S by R. Just as $a\downarrow''\beta$ is the class of ordinal couples $a\downarrow x$ such that x is in β (65.6), so also $a\downarrow;S$ is the relation between two ordinal couples $a\downarrow x$ and

$\alpha \downarrow y$ such that x bears S to y. Similarly $\downarrow b; S$ is the relation between two ordinal couples $x \downarrow b$ and $y \downarrow b$ such that x bears S to y.

66.28. Just as $`\alpha Q b`$ is an abbreviation for $`Ob``\alpha`$ (65.7), so also $`S Q b`$ is an abbreviation for $`Ob; S`$ ([PM] *150.03), and in particular $`S \downarrow b`$ is an abbreviation for $`\downarrow b; S`$. Also, just as $`\alpha \downarrow `` \beta`$ denotes the class of those classes $\downarrow y`` \alpha$ where $y$ is in $\beta$ (and where $\overset{\cdot\cdot}{\downarrow} y`` \alpha$ is itself the class of those ordinal couples $x \downarrow y$ such that x is in α), so also $`S \downarrow ; T`$ denotes the relation of a relation $\downarrow y; S$ to a relation $\downarrow z; S$ such that y bears T to z.

66.29. $`R \dagger`$ for $`R;`$. Compare with $`R_c`$ (66.17). This symbol '$\dagger$' as used here is not to be confused with '$\dagger$' as defined in 61.4.

66.30. $`R \parallel S`$ for $`(R|)|(|S)`$ ([PM] *43.01). Thus $R \parallel S$ is the relation of P to Q such that $P = R|Q|S$ (61.3), and $[R \parallel S]`Q$ is the same relation as $R|Q|S$. Furthermore, $[R \parallel \breve{R}]`Q$ is the same relation as $R; Q$ (66.27), and $R \parallel \breve{R}$ is the same relation as $R\dagger$ (66.29).

66.31. Suppose we have given a class κ of classes, and we wish a relation R which relates exactly one member of each member α of κ to that member α, so that $R`\alpha$ can be called the *selected member* of $\alpha$, and $R$ can be called a *selective relation* for $\kappa$. The required $R$ would have to be such that $R \in [1 \to \mathrm{Cls}] . [R \subset \in] . [\mathrm{Cl}`R = \kappa]$ (64.31, 63.7, 66.13, 66.02, 60.46), that is, a one-many relation, a subrelation of $\in$, and such that κ is its converse domain. The class of all such selective relations for κ may be denoted by $`\in_{\Delta}`\kappa`$, where $`\in_{\Delta}`$ is the relation of the class $\mu$ of selective relations for $\kappa$ to $\kappa$ itself, and where $`\in_{\Delta}`$ abbreviates the expression $`\hat{\mu}\hat{R}[\mu = \hat{R}[R \in [1 \to \mathrm{Cls}]] . [R \subset \in] . [\mathrm{Cl}`R = \kappa]]`$. More generally, given any relation P which has κ as its converse domain, we define $`P_{\Delta}`$ as $`\hat{\mu}\hat{R}[\mu = \hat{R}[R \in [1 \to \mathrm{Cls}]] . [R \subset P] . [\mathrm{Cl}`R = \kappa]]`$ and we call $P_{\Delta}`\kappa$ the class of *P-selections* from κ ([PM] *80). Each P-selection from κ is a one-may subrelation of P that has κ as its converse domain. Such a P-selection from κ relates exactly one member of each class of the form $\vec{P}`y$ to $y$ itself, where $y$ is a member of $\kappa$. If $P$ is the relation $\in$, then $y$ must be a class $\alpha$, and $\vec{P}`y$ must be the same as α. In this case a P-selection from κ is a selective relation for κ.

66.32. The symbol '$\ddagger$' is defined ([PM] *85.5) in such a way that $R \ddagger y =$ $\downarrow y`` \vec{R}`y$ (cp. 63.4, 64.24, 65.4, 66.08), so that $R \ddagger y$ is the class of ordinal couples having the things that bear R to y as first elements and having y itself as a second element. In particular, $\in \ddagger \alpha$ is the class of ordinal couples that have the members of α as first elements and α itself as a second element. Thus $\in \ddagger \alpha$ and $\in \ddagger \beta$ are non-overlapping classes if α and β are distinct, and they respectively have the same cardinal numbers as α and β. (Cp. 81.)

67. Relations of more than two terms

67.1. We can view relations of more than two terms as propositional functions of more than two arguments (05.9). Just as $R(a, b)$ may be considered to be a proposition asserting that a two-term relation R relates a to b, so also $S(a, b, c)$ may be considered to be a proposition asserting that a three-termed relation S relates a, b, and c (in that order); and $T(a, b, c, d)$ may similarly be considered to be a proposition asserting that a four-termed relation T relates a, b, c, and d (in that order); and so on for relations of any finite number of terms.

67.2. Just as $`R(a, b)`$ is sometimes written as $`aRb`$, where $`R`$ denotes a two-termed relation, so also $`S(a, b, c)`$ may be written as $`aSb, c`$ or as $`aS(b, c)`$, where $`S`$ denotes a three-termed relation. (This sort of notation is used in Hull, Hovland, Ross, Hall, Perkins, Fitch, II 37.)

67.3. Just as two-termed relations may be viewed as classes of ordered couples (60.1), so also three-termed relations may be viewed as classes of ordered triples, and four-termed relations as classes of ordered quadruples, and so on. If the notation $`\langle a, b \rangle`$ is used to denote the ordered couple a, b (60.31, 64.26), then $`\langle a, b, c \rangle`$ can be used to denote the ordered triple a, b, c, where $`\langle a, b, c \rangle`$ can be treated as an abbreviation for $`\langle \langle a, b \rangle, c \rangle`$ ([R] 283). Similarly, $`\langle a, b, c, d \rangle`$ may be used to denote an ordered quadruple and may be treated as an abbreviation for $`\langle \langle a, b, c \rangle, d \rangle`$; and so on. In Gödel VI 112, a similar notation is used but with omission of commas and with the convention that $`\langle abc \rangle`$ abbreviates $`\langle a \langle bc \rangle \rangle`$, and $`\langle abcd \rangle`$ abbreviates $`\langle a \langle bcd \rangle \rangle`$, and so on.

67.4. If the ordered couple a, b is denoted by $`a; b`$ (60.31), then the

ordered triple a, b, c may be denoted by $'a;b;c'$, where $'a;b;c'$ abbreviates $'(a;b);c'$. This same method can be extended to ordered quadruples, ordered quintuples, and so on.

67.5. An example of a three-termed relation is the relation of jealousy between a, b, and c such that a is jealous of b because of c. If 'S' denotes this relation, then the proposition that a is jealous of b because of c could be expressed in any of the following ways:

$$S(a, b, c),$$
$$aSb, c,$$
$$aS(b, c),$$
$$\langle a, b, c\rangle \in S,$$
$$\langle abc\rangle \in S,$$
$$a;b;c \in S.$$

CHAPTER 7

ARITHMETIC FORMALIZED AS AN INDEPENDENT DISCIPLINE

70. Introduction. By arithmetic in logical writings we normally mean the arithmetic of the non-negative integers or natural numbers, 0, 1, 2, ... The arithmetic of natural numbers can be developed either as an independent discipline or as part of a wider underlying theory such as set theory in one of its forms, or combinatory logic. The first course (and the one considered in this chapter) is followed by [HB] and [K], the latter by [Q], [R], and Gödel (4183), after the model of [PM], and by Rosser (5461) (43.5, 45.4).

71. The Peano axioms. Following methods that approximate to those used originally by Peano (717), the properties of the natural numbers can be described in axioms whose primitive terms are:

(1) 'N', denoting the set of natural numbers;
(2) the variables 'x', 'y', 'z',... for integers, and '0' for zero;
(3) a symbol for the concept of successor, where the *successor* of an integer is the integer one greater. The prime symbol ''' is often used for this purpose. Thus '0'' denotes the number 1, '0'' denotes the number 2, and so on. In general 'n'' denotes the successor of n.

The following then correspond to Peano's axioms, where '$=$' is used for the intuitive notion of equality and where (v) expresses the principle of mathematical induction:

(i) $0 \in \mathbf{N}$,
(ii) $(x)[x \in \mathbf{N} \supset x' \in \mathbf{N}]$,
(iii) $(x)(y)[[x \in \mathbf{N} . y \in \mathbf{N} . x' = y'] \supset x = y]$,
(iv) $(x)[x \in \mathbf{N} \supset 0 \neq x']$,
(v) $[\phi 0 . (x)[[x \in \mathbf{N} . \phi x] \supset \phi x']] \supset (x)[x \in \mathbf{N} \supset \phi x]$.

72. Formulation of the theory as an independent discipline

72.1. Consider now the theory of natural numbers developed as an independent discipline. All variables are taken as ranging over (natural) numbers, so that axioms (i) and (ii) can be omitted and the qualifications '$x \in \mathrm{N}$' and '$y \in \mathrm{N}$' are dropped throughout (iii) and (v). Such a development is normally carried out in what is called an elementary manner, *i.e.*, in a formalism without free (*a fortiori* without bound) predicate variables. Axiom (v) is therefore treated as an axiom schema, and 'ϕ' stands for an arbitrary context. This yields the following formalization:

72.2. Terms are '0', variables, and everything of the form 't'' where 't' is a term. Expressions '0', '0'', '0'',..., are numerals.

72.3. Atomic formulas are of the form '$r = s$', where 'r' and 's' are terms. Formulas are built up from atomic formulas by means of truth-functional connectives and quantifiers in the customary fashion. Axioms are all those formulas (in the sense just defined) which are instances of the axioms of some chosen formulation of the first-order functional calculus with identity (25), or of (iii), (iv) or (v) of 71 as formulated according to 72.1. Rules of inference are those of the chosen system of first-order functional calculus.

73. Recursive definition of addition and multiplication

73.1. Elementary number theory is usually taken to include more than the notations and notions included in sections 71 and 72. The study of addition and multiplication is also included, with the customary arithmetical notation for these two operations. Thus the definition of elementary arithmetic given above is to be extended as follows:

73.2. Terms are extended to include the expressions '$r + s$' and '$r \times s$', (or 'rs'), where 'r' and 's' are terms. Formulas are built up for the extended class of terms in the customary way. Axioms comprise those already given together with axioms which are the so-called *recursive definitions* of addition and multiplication:

73.3. $x + 0 = x$,
$x + y' = (x + y)'$.

73.4. $x0 = 0$,
$xy' = xy + x$.

73.5. The two classical formulations of axiomatic elementary number theory, [HB] and [K], both have the above axioms superimposed on a basis of the first-order functional calculus with identity. (Universal quantifiers could be thought of as attached to the above axioms, but this is not necessary if suitable rules of inference are used.)

73.6. The recursive definitions form a genuine addition to the expressive power of the theory. These definitions also enable us to eliminate the signs for addition and multiplication from any term in which, in addition to these signs, only constant numerals (built up out of '0' and ''') occur. However, these definitions do not always enable us to eliminate the signs for addition and multiplication if variables are present.

74. **Recursive arithmetic.** Recursive arithmetic is of special interest to the logician as an auxiliary theory used in metamathematics (Chapter 9). As our concern is with the explanation of logical symbols, we refrain from giving here the definitions of the various forms of recursion, and we refrain from giving the recursive definitions of the more familiar recursive mathematical functions (with the exception, of course, of addition and multiplication, given above). Among the functions thus omitted are exponentiation and the factorial function.

75. Auxiliary concepts

75.1. For the recursive definition of inverse operations (analogues of subtraction and division) we need some auxiliary concepts.

75.2. '$\mathrm{pd}(n)$' may be interpreted as the *predecessor* of n, provided that 0 is its own "predecessor" ([K] 223).

$$\mathrm{pd}(0) = 0.$$
$$\mathrm{pd}(x') = x.$$

The notation of [HB] for '$\mathrm{pd}(n)$' is '$\delta(n)$' (i 300).

ordered triple a, b, c may be denoted by $`a;b;c'$, where $`a;b;c'$ abbreviates $`(a;b),c'$. This same method can be extended to ordered quadruples, ordered quintuples, and so on.

67.5. An example of a three-termed relation is the relation of jealousy between a, b, and c such that a is jealous of b because of c. If $`S'$ denotes this relation, then the proposition that a is jealous of b because of c could be expressed in any of the following ways:

$$S(a, b, c),$$
$$aSb, c,$$
$$aS(b, c),$$
$$\langle a, b, c\rangle \in S,$$
$$\langle abc\rangle \in S,$$
$$a;b;c \in S.$$

CHAPTER 7

ARITHMETIC FORMALIZED AS AN INDEPENDENT DISCIPLINE

70. Introduction. By arithmetic in logical writings we normally mean the arithmetic of the non-negative integers or natural numbers, 0, 1, 2, ... The arithmetic of natural numbers can be developed either as an independent discipline or as part of a wider underlying theory such as set theory in one of its forms, or combinatory logic. The first course (and the one considered in this chapter) is followed by [HB] and [K], the latter by [Q], [R], and Gödel (41*B*), after the model of [PM], and by Rosser (54*A*) (43.5, 45.4).

71. The Peano axioms. Following methods that approximate to those used originally by Peano (71*A*), the properties of the natural numbers can be described in axioms whose primitive terms are:

(1) '$\mathbf{N}$', denoting the set of natural numbers;
(2) the variables 'x', 'y', 'z',... for integers, and '0' for zero;
(3) a symbol for the concept of successor, where the *successor* of an integer is the integer one greater. The prime symbol "'" is often used for this purpose. Thus '$0'$' denotes the number 1, '$0''$' denotes the number 2, and so on. In general 'n'' denotes the successor of n.

The following then correspond to Peano's axioms, where '$=$' is used for the intuitive notion of equality and where (v) expresses the principle of mathematical induction:

(i) $0 \in \mathbf{N}$,
(ii) $(x)[x \in \mathbf{N} \supset x' \in \mathbf{N}]$,
(iii) $(x)(y)[[x \in \mathbf{N} . y \in \mathbf{N} . x' = y'] \supset x = y]$,
(iv) $(x)[x \in \mathbf{N} \supset 0 \neq x']$,
(v) $[\phi 0 . (x)[[x \in \mathbf{N} . \phi x] \supset \phi x']] \supset (x)[x \in \mathbf{N} \supset \phi x]$.

72. Formulation of the theory as an independent discipline

72.1. Consider now the theory of natural numbers developed as an independent discipline. All variables are taken as ranging over (natural) numbers, so that axioms (i) and (ii) can be omitted and the qualifications '$x \in \mathrm{N}$' and '$y \in \mathrm{N}$' are dropped throughout (iii) and (v). Such a development is normally carried out in what is called an elementary manner, i.e., in a formalism without free (*a fortiori* without bound) predicate variables. Axiom (v) is therefore treated as an axiom schema, and 'ϕ' stands for an arbitrary context. This yields the following formalization:

72.2. Terms are '0', variables, and everything of the form 't'' where 't' is a term. Expressions '0', '0'', '0''',..., are numerals.

72.3. Atomic formulas are of the form '$r=s$', where 'r' and 's' are terms. Formulas are built up from atomic formulas by means of truth-functional connectives and quantifiers in the customary fashion. Axioms are all those formulas (in the sense just defined) which are instances of the axioms of some chosen formulation of the first-order functional calculus with identity (25), or of (iii), (iv) or (v) of 71 as formulated according to 72.1. Rules of inference are those of the chosen system of first-order functional calculus.

73. Recursive definition of addition and multiplication

73.1. Elementary number theory is usually taken to include more than the notations and notions included in sections 71 and 72. The study of addition and multiplication is also included, with the customary arithmetical notation for these two operations. Thus the definition of elementary arithmetic given above is to be extended as follows:

73.2. Terms are extended to include the expressions '$r+s$' and '$r \times s$', (or 'rs'), where 'r' and 's' are terms. Formulas are built up for the extended class of terms in the customary way. Axioms comprise those already given together with axioms which are the so-called *recursive definitions* of addition and multiplication:

73.3. $x+0=x$,
$x+y'=(x+y)'$.

73.4. $x0=0$,
$xy'=xy+x$.

73.5. The two classical formulations of axiomatic elementary number theory, [HB] and [K], both have the above axioms superimposed on a basis of the first-order functional calculus with identity. (Universal quantifiers could be thought of as attached to the above axioms, but this is not necessary if suitable rules of inference are used.)

73.6. The recursive definitions form a genuine addition to the expressive power of the theory. These definitions also enable us to eliminate the signs for addition and multiplication from any term in which, in addition to these signs, only constant numerals (built up out of '0' and ''') occur. However, these definitions do not always enable us to eliminate the signs for addition and multiplication if variables are present.

74. Recursive arithmetic. Recursive arithmetic is of special interest to the logician as an auxiliary theory used in metamathematics (Chapter 9). As our concern is with the explanation of logical symbols, we refrain from giving here the definitions of the various forms of recursion, and we refrain from giving the recursive definitions of the more familiar recursive mathematical functions (with the exception, of course, of addition and multiplication, given above). Among the functions thus omitted are exponentiation and the factorial function.

75. Auxiliary concepts

75.1. For the recursive definition of inverse operations (analogues of subtraction and division) we need some auxiliary concepts.

75.2. '$\mathrm{pd}(n)$' may be interpreted as the *predecessor* of n, provided that 0 is its own "predecessor" ([K] 223).

$$\mathrm{pd}(0)=0,$$
$$\mathrm{pd}(x')=x.$$

The notation of [HB] for '$\mathrm{pd}(n)$' is '$\delta(n)$' (1 300).

75.3. $sg(n)$ is 0 if $n=0$ and 1 if $n>0$ ([K] 223).

75.4. $\overline{sg}(n)$ is 0 if $n>0$ and 1 if $n=0$ ([K] 223).

75.5. $\min(a, b)$ is the least of two numbers a and b ([K] 223). More precisely, $\min(a, b)$ is exactly $b \dot{-} (b \dot{-} a)$ ([K] 223), where $a \dot{-} b$ is defined as in 76.2.

75.6. $\max(a, b)$ is the greater of the numbers a and b ([K] 223). More precisely, $\max(a, b)$ is exactly $(a+b) \dot{-} \min(a, b)$ ([K] 223).

76. Inverse operations

76.1. Analogues to subtraction and division must be defined. The values of these operations agree with ordinary arithmetic except where the values would not be natural numbers (non-negative integers) in ordinary arithmetic. These operations are so defined that their values are natural numbers in all cases, even where they would not be such in ordinary arithmetic.

76.2. $a \dot{-} n$ is the difference of a and n if $a \geqslant n$, and is 0 otherwise ([K] 223). In terms of the pd function (75.2),

$$x \dot{-} 0 = x,$$
$$x \dot{-} y' = \mathrm{pd}(x \dot{-} y).$$

The notation of [HB] for '$a \dot{-} n$' is '$\delta(a, n)$' (i 303).

76.3. The absolute value $|a-n|$ of the arithmetic difference of a and n is $(a \dot{-} n) + (n \dot{-} a)$ ([K] 223).

76.4. The remainder of the division of a by b, namely $\mathrm{rm}(a, b)$, is defined thus:

$$\mathrm{rm}(0, y) = 0,$$
$$\mathrm{rm}(x', y) = (\mathrm{rm}(x, y))' \, \mathrm{sg}(|y-(\mathrm{rm}(x, y))'|) \quad ([K] 223).$$

The notation of [HB] for '$\mathrm{rm}(a, b)$' is '$\rho(a, b)$' (i 317).

76.5. The quotient of a by b, namely a/b is defined as follows:

$$0/y = 0,$$
$$x'/y = x/y + \overline{sg}(|y-(\mathrm{rm}(x, y))'|) \quad ([K] 223).$$

77. The μ-function

77.1. The function whose value for argument z is expressed by $\mu y_{y<z} R(y)$ has as its value the least y such that $y<z$ and $R(y)$ if there is such a y; and otherwise the function has the value z ([K] 225). The notation corresponding to this Kleene notation in Péter (XVI 280) is '$\mu_i[i<n \,\&\, R(y)]$'. If there is a y such that $R(y)$, then the expression $\mu y R(y)$ denotes the least such y ([K] 279). If there is no such y, the expression has no denotation.

77.2. By means of the μ-function, expressions can be defined such as '$\varepsilon y R(x, y)$', which denotes the least y such that $R(x, y)$ if there is such a y, and denotes 0 otherwise ([K] 317).

CHAPTER 8

NUMBERS AS DEFINED WITHIN SYSTEMS OF LOGIC

80. Introduction. If arithmetic is treated as an independent discipline, the concept of number is usually taken as undefined; but if arithmetic is treated as part of a system of logic, the concept of number may be definable in terms of other concepts. In particular the concepts *zero* and *successor* may be definable in terms of other concepts. In any case, the definitions used must be such as to render the Peano axioms (71) provable. The definitions are different in different systems of logic, but two main traditions are discernible. The first (81), followed by [Q] and [R], stems via [PM] from Frege's definitions of a number as a class of similar classes. The second tradition (82) involves the theory of sets as developed by Zermelo, Fraenkel, Bernays, von Neumann, and Gödel. A third method, used in combinatory logic, has been indicated in 43.5.

81. The definitions of cardinal numbers by similarity

81.1. Two classes α and β are called *equinumerous* or *similar* if there exists a one-to-one correspondence between the members of α and those of β, that is, if there exists a relation R which is one-one (64.33) and has α for its domain (63.6) and β for its codomain (63.7), so that $\alpha = R``\beta$ (63.4). Such a relation R is called a cardinal correlator of α with β. The class $\alpha \dot{\mathrm{sm}} \beta$ is the class of cardinal correlators of α with β([PM] *73.01). We may regard '$\alpha \dot{\mathrm{sm}} \beta$' as an abbreviation for '$\hat{R}[[R \in 1 \to 1] . [\alpha = D`R] . [\beta = \breve{C}`R]]$'. [R] (345) writes '$\alpha \ \mathrm{sm}_R \ \beta$' to assert that R is a cardinal correlator of α with β.

81.2. α and β are *similar*, α sm β ([PM] *73.02), if $\exists ! \alpha \dot{\mathrm{sm}} \beta$ (53.6, [PM] *73.04), that is, if there exists a cardinal correlator of α with β.

81.3. Frege – and following him Russell – treat the *cardinal number* of

a class as the class of all classes similar to that class. Thus we can regard Nc'α (the cardinal number of α) as $\hat{\beta}[\beta \ \mathrm{sm} \ \alpha]$ ([PM] *101.1), and NC (the class of cardinal numbers) as $\hat{\alpha}(\exists \beta)[\alpha = \mathrm{Nc}`\beta]$, that is, as D'Nc ([PM] *100.02). These are essentially the definitions of Frege, [Q], and [R]. The notations 'Nc' and 'NC' occur in [PM], in Quine VII 121, and in [R] (371–372), [R] writing '$\mathrm{Nc}(\alpha)$' instead of '$\mathrm{Nc}`\alpha$' (64.42, 66.13).

81.4. The notion of cardinal number is broader than that of natural number. A natural number is a finite ("inductive") cardinal number in the sense of [PM] *120 (cp. [R] 390). The theory of numbers in [PM] is developed for cardinal numbers in all their generality (cp. [PM] *100) and also for relation-numbers (cp. 85 and [PM] *152).

81.5. If within this general approach of Russell, Whitehead, Quine, and Rosser we wish to develop separately a theory of natural numbers, we must define the number 0, the successor function, and the class of natural numbers (finite cardinal numbers).

81.51. 0 may be defined as the cardinal number of the empty class, that is, as the class of all classes similar to the empty class. Since nothing is similar to the empty class but the empty class itself, 0 may be defined more simply as the class whose only member is the empty class. Thus '0' for '$\iota`\Lambda$' (64.21, [Q] 237) or for '$\{\Lambda\}$' ([R] 252).

81.52. The definition of successor is motivated as follows. If a class has n members, then a class β which contains all members of α and in addition one more member will contain $n+1$ members. Inversely, if a class α is obtained from a class β by omitting one of the members of β, then α will have one less member than β. Using the notation '$\mathrm{S}`n$' (cp. [Q] 238) for *the successor of* n, we can define 'S' in such a way that

$$\mathrm{S}`n = \hat{\beta}(\exists y)[y \in \beta . [(\beta \cap -\{y\}) \in n]] .$$

In the above discussion the classes α and β are assumed to be finite.

81.53. A natural number is either 0, or the successor of 0, or the successor of the successor of 0, and so on. That is, a natural number is

CHAPTER 8

NUMBERS AS DEFINED WITHIN SYSTEMS OF LOGIC

80. Introduction. If arithmetic is treated as an independent discipline, the concept of number is usually taken as undefined; but if arithmetic is treated as part of a system of logic, the concept of number may be definable in terms of other concepts. In particular the concepts *zero* and *successor* may be definable in terms of other concepts. In any case, the definitions used must be such as to render the Peano axioms (71) provable. The definitions are different in different systems of logic, but two main traditions are discernible. The first (81), followed by [Q] and [R], stems via [PM] from Frege's definitions of a number as a class of similar classes. The second tradition (82) involves the theory of sets as developed by Zermelo, Fraenkel, Bernays, von Neumann, and Gödel. A third method, used in combinatory logic, has been indicated in 43.5.

81. The definitions of cardinal numbers by similarity

81.1. Two classes α and β are called *equinumerous* or *similar* if there exists a one-to-one correspondence between the members of α and those of β, that is, if there exists a relation R which is one-one (64.33) and has α for its domain (63.6) and β for its codomain (63.7), so that $\alpha = R``\beta$ (63.4). Such a relation R is called a cardinal correlator of α with β. The class $\alpha\overline{\text{sm}}\beta$ is the class of cardinal correlators of α with β ([PM] *73.01). We may regard '$\alpha\overline{\text{sm}}\beta$' as an abbreviation for '$\hat{R}[[R \in 1 \to 1] . [\alpha = D`R] . [\beta = \breve{C}`R]]$'. [R] (345) writes '$\alpha\ \text{sm}_R\ \beta$' to assert that R is a cardinal correlator of α with β.

81.2. α and β are *similar*, α sm β ([PM] *73.02), if $\exists!\alpha\overline{\text{sm}}\beta$ (53.6, [PM] *73.04), that is, if there exists a cardinal correlator of α with β.

81.3. Frege – and following him Russell – treat the *cardinal number* of

a class as the class of all classes similar to that class. Thus we can regard Nc`$\alpha$ (the cardinal number of $\alpha$) as $\hat{\beta}[\beta \text{ sm } \alpha]$ ([PM] *101.1), and NC (the class of cardinal numbers) as $\hat{\alpha}(\exists\beta)\{\alpha = \text{Nc}`\beta\}$, that is, as D`Nc ([PM] *100.02). These are essentially the definitions of Frege, [Q], and [R]. The notations 'Nc' and 'NC' occur in [PM], in Quine VII 121, and in [R] (371–372), [R] writing '$\text{Nc}(\alpha)$' instead of '$\text{Nc}`\alpha$' (64.42, 66.13).

81.4. The notion of cardinal number is broader than that of natural number. A natural number is a finite ("inductive") cardinal number in the sense of [PM] *120 (cp. [R] 390). The theory of numbers in [PM] is developed for cardinal numbers in all their generality (cp. [PM] *100) and also for relation-numbers (cp. 85 and [PM] *152).

81.5. If within this general approach of Russell, Whitehead, Quine, and Rosser we wish to develop separately a theory of natural numbers, we must define the number 0, the successor function, and the class of natural numbers (finite cardinal numbers).

81.51. 0 may be defined as the cardinal number of the empty class, that is, as the class of all classes similar to the empty class. Since nothing is similar to the empty class but the empty class itself, 0 may be defined more simply as the class whose only member is the empty class. Thus '0' for '$\iota`\Lambda$' (64.21, [Q] 237) or for '$\{\Lambda\}$' ([R] 252).

81.52. The definition of successor is motivated as follows. If a class has n members, then a class β which contains all members of α and in addition one more member will contain $n+1$ members. Inversely, if a class α is obtained from a class β by omitting one of the members of β, then α will have one less member than β. Using the notation '$\text{S}`n$' (cp. [Q] 238) for *the successor of n*, we can define 'S' in such a way that

$$\text{S}`n = \hat{\beta}(\exists y)\{y \in \beta . [[\beta \cap -\{y\}] \in n]\}.$$

In the above discussion the classes α and β are assumed to be finite.

81.53. A natural number is either 0, or the successor of 0, or the successor of the successor of 0, and so on. That is, a natural number is

anything which stands in the ancestral (cp. 66.19, [PM] *90) of the successor relation to 0. This definition is common to Frege, Russell, [Q], and [R].

81.6. In [PM] and in general in systems based on the theory of types (31), a complication arises in the given definitions. For though they can be carried out verbatim in each type, one obtains thus a whole hierarchy of number systems, one in each type (beginning with type $o(o\iota)$ (cp. 32)). Thus 2 in type $o(o\iota)$ is the class of all two-membered classes of individuals, and so on. This duplication (which, while it does not invalidate any of the arguments of classical mathematics, certainly complicates them considerably) was early felt as a disadvantage in the theory of types, and it was a primary motivation in the evolution of Quine's two systems (4585 and [Q]). [PM] has a complicated system of notation for the relative types of numbers. We shall not deal with it here (cp. [PM] *12, *103–*106).

82. Von Neumann's method of defining numbers

82.1. The second tradition is that of set theory as developed by Zermelo, Fraenkel, Bernays, von Neumann, and Gödel. Within this second tradition the natural numbers are defined by a method different from that described above. Here a special technical meaning of the term *set* (German, *Menge*) plays an important rôle (cp. 50.2), and this is why, in this connection, it is usual to speak of *set theory*. All sets, in this sense, are classes, but not all classes are sets. More specifically, sets are those classes which are members of other classes. Classes which are not themselves members of classes are said to be *proper classes*. Sets, then, are those classes which are not proper classes. The class of all sets is an example of a proper class. According to this view, there is no class of all classes.

82.2. If, in set theory (as developed by Bernays and Gödel), we attempt to define the number n as the class of all n-numbered sets, then the number n would turn out not to be a set itself but to be a proper class. Using a definition of number due to von Neumann, however, numbers themselves can be treated as sets. This method consists in identifying 0 with the empty set, 1 with the set that has 0 as its only member, 2 with the set that has 0 and 1 as its only members, and so on. In general, each

number is defined to be the set of its predecessors. The effect of this is that each number n is a set of exactly n members. Thus '0' can be defined as 'Λ' (51.4) and the symbol 'S' for the successor function can be defined in such a way that $S'n = n \cup \{n\}$. The set of all natural numbers can then be defined as in 81.53, but this set, from the present standpoint, is itself a transfinite number (though not a natural number), and in fact the smallest transfinite number ω. This method is applicable not only to definitions of transfinite numbers but is quite convenient in dealing with transfinite induction. (Cp. Gödel VI 112.)

82.3. It should be noted that if the method of 81.3 is used, then to say that a class has n members is to say that the class is a member of the class which is the number n, while if the method of 82.2 is used, then to say that a set has n members is to say that the set is similar (in the sense of 81.1) to the set which is the number n.

83. Arithmetic operations on natural numbers, defined by means of the successor

83.1. We were obliged to take addition and multiplication as undefined notions in the formalization of arithmetic as an independent discipline (72), but a formal system in which zero, the successor function, and the class of natural numbers can be defined is also normally sufficiently strong to provide definitions of addition and multiplication of natural numbers. There are two ways in which this can be done, and each of these ways may be combined with either of the two ways of introducing numbers described in sections 81 and 82 above. The first of the two methods is that followed by Quine ([Q] 259) and applies only to inductive (finite) cardinal numbers. The second method (84 below) used by Whitehead and Russell ([PM] *110–*116) applies to all cardinal numbers, whether finite or infinite.

83.2. The method followed by [Q] (253 ff.) starts from a definition of *the nth power of the relation R*. We have already defined the relative powers R^1, R^2, R^3, and so on (61.5). But these definitions do not presuppose that '2', '3', and so on, denote the cardinal numbers 2, 3, and so on. On the other hand, to obtain a general definition of R^n where n is any finite (inductive) cardinal number, use can be made of the method of

[PM] *301, or of the simpler method of [Q] 255, involving the auxiliary notion, $\flat R$. The latter method will now be outlined.

83.3. Let us consider the pairs $a;0$, $b;1$, $c;2$, ... (cp. 60.31), corresponding intuitively to "a with number 0", "b with number 1", "c with number 2", and so on, where a, b, c, and so on are members of the field of R (63.8). The auxiliary relation $\flat R$ ([Q] 255) is defined as the relation of any pair $x;n$ to any pair $y;m$, such that xRy and $m = \mathrm{S}'n$ are both true. Then R^n can be defined as the relation of anything x to anything y such that the pair $x;0$ bears the ancestral $*\flat R$ (66.19) of $\flat R$ to the pair $y;n$. Thus $'R^n{}'$ is an abbreviation for $'\hat{x}\hat{y}(*\flat R(x;0,\ y;n))'$ ([Q] 255).

83.4. The definitions of sum, product, and power of natural numbers follow as closely as possible those of ordinary elementary arithmetic. They are as follows:

83.5. $m+n$ is the number which bears to m the nth relative power of the successor relation S to m. Thus $'m+n'$ is an abbreviation for $'\mathrm{S}^{n}{}'m'$ ([Q] 259).

83.6. $m \times n$ is the number obtained from 0 by n additions of m. Thus, $'m \times n'$ is an abbreviation for $'(\lambda_a[m+a])^{n}{}'0'$ (65.2, [Q] 259).

83.7. $m \wedge n$, that is, m^n, is the number obtained from 1 by n multiplications by m. (1 is defined as S'0.) Thus $'m \wedge n'$ is an abbreviation for $'(\lambda_a[m \times a])^{n}{}'1'$ (65.2, [Q] 259). The symbol '$\wedge$' is an inverted radical sign. $m \wedge n$ (or m^n) is the nth power of the natural number m, whereas R^n is the nth relative power of the relation R.

84. Arithmetic operations on cardinal numbers

84.1. The definitions of addition, multiplication, and exponentiation in [PM] and [R] are all framed in the same general way, and they apply to infinite as well as to finite (inductive) cardinal numbers. In each case an operation on classes is first defined, rather than the desired arithmetic operation on cardinal numbers. The operation on classes is so chosen that if n and m are the cardinal numbers respectively of α and β, then the operation on the classes α and β gives a class whose cardinal number is the result of the desired arithmetic operation on n and m.

84.2. In the case of addition of cardinal numbers, we can let '$+$' stand for the arithmetic addition of classes, the operation on classes referred to in 84.1, and '$+_c$' stand for arithmetic addition of cardinal numbers, the corresponding operation on the cardinal numbers themselves ([PM] *110.01, *110.02). One way to define $'[\alpha+\beta]'$ is as an abbreviation for $'[\alpha \times \{\Lambda\}] \cup [\beta \times \{\mathrm{V}\}]'$ (54.6, 51.4, 64.21). Then if α has n members and β has m members, $\alpha+\beta$ will have the sum of n and m as the number of its members, even if α and β have members in common. (This is essentially the procedure used in [R] 374.) On the other hand, $\alpha \cup \beta$ has the sum of n and m as the number of its members only if α and β have no member in common. (The present use of '$+$' should not be confused with '$+$' as meaning the same as '$\cup$' as in 54.4, or with '$+$' as meaning addition in the ordinary mathematical sense (as in 73.2).) In [PM] *110.01 a different and somewhat more complicated meaning is assigned to $'[\alpha+\beta]'$, but $'[\alpha+\beta]'$, as there defined, serves the same purpose of denoting a class having as many members as the sum of n and m, where α has n members and β has m members. The definition of $'[\alpha+\beta]'$ used in [PM] *110.01 is $'[\downarrow[\Lambda \cap \beta]\text{``}\iota\text{``}\alpha] \cup [(\Lambda \cap \alpha]\downarrow\text{``}\iota\text{``}\beta]'$. Thus $[\alpha+\beta]$ is here being treated as the union of two classes of ordinal couples, the first class consisting of ordinal couples each of which has a unit class of a member of α as its first element and the empty class, $[\Lambda \cap \beta]$, of the same type as β, as its second element, and the second class consisting of ordinal couples each of which has the empty class, $[\Lambda \cap \alpha]$, of the same type as α, as its first element and a unit class of a member of α as its second element. The first class has as many members as α has, and the second class has as many members as β has; and the two classes are non-overlapping, so $[\alpha+\beta]$ has the desired number of members.

84.3. The arithmetic sum of two cardinals will be denoted by $'\mu +_c \nu'$ (the 'c' standing for "cardinal") which is defined as $'\hat{\xi}(\exists\alpha)(\exists\beta)[[\mu = \mathrm{Nc}'\alpha] \cdot [\nu = \mathrm{Nc}'\beta] \cdot [\xi = \mathrm{Nc}'[\alpha+\beta]]]'$ ([PM] *110.02). Thus if $\mu = \mathrm{Nc}'\alpha$ and $\nu = \mathrm{Nc}'\beta$, then $[\mu +_c \nu] = \mathrm{Nc}'[\alpha+\beta]$ (cp. 81.2, 81.3). (In the foregoing sentences and hereafter, the Greek letters 'γ', 'μ', 'ν', 'ξ' are used like italic letters for metalinguistic purposes. See 08.3–08.8 and especially 08.7.)

84.4. The definition of 84.2 can be used only for adding a finite number of classes. For adding an infinity of classes a more elaborate kind of addition must be defined, applying to any finite or infinite class κ of classes. $\Sigma'\kappa$ is called the *arithmetic sum of the class of classes* κ and is a class whose cardinal number is the arithmetic sum of the cardinal numbers of the members of κ. 'Σ' is defined in such a way that $\Sigma'\kappa = s'[\epsilon \downarrow ''\kappa]$ ([PM] *112.01), and so that '$\Sigma'\kappa$' denotes the class of all ordinal couples of the form $x \downarrow \alpha$ such that α is a member of κ and x is a member of α. (Cp. 66.32.) Clearly the cardinal number of $\Sigma'\kappa$ is the sum of the cardinal numbers of members of κ. '$\Sigma Nc'\kappa$' abbreviates '$Nc'\Sigma'\kappa$' and denotes the sum of the cardinal numbers of the classes which are members of κ.

84.5. The arithmetic product of two classes α and β is written as '$\beta \times \alpha$', which is defined as $s'[\alpha \downarrow ''\beta]$ in [PM] *113.02. This is the class of all ordinal couples such that the first element is in α and the second element is in β(65.7), so that the number of members of $\beta \times \alpha$ is the product of the number of members of α with the number of members of β. (In [PM] *113, the notation '$\beta \times \alpha$' is used instead of '$\alpha \times \beta$' because of certain analogies with products in relation arithmetic.) (See 08.9 with respect to 's'.)

84.6. The arithmetic product of two cardinals will be denoted by '$\mu \times_c \nu$', which is defined as '$\hat{\xi}(\exists\alpha)(\exists\beta)[[\mu = Nc'\alpha].[\nu = Nc'\beta].[\xi \text{ sm } [\alpha \times \beta]]]$' ([PM] *113.03). Thus if $\mu = Nc'\alpha$ and $\nu = Nc'\beta$, then $[\mu \times_c \nu] = Nc'[\alpha \times \beta]$.

84.7. The definition of 84.5 can be used for multiplication of only a finite number of classes. For multiplying an infinity of classes a more elaborate kind of multiplication must be defined, applying to any finite or infinite class κ. We may regard $\epsilon_\Delta'\kappa$, the class of selective relations of κ (66.31), as the required arithmetic product of the class of classes κ. It can be shown that the cardinal number of $\epsilon_\Delta'\kappa$ is the arithmetic product of the cardinal numbers of the members of κ. '$\Pi Nc'\kappa$' abbreviates '$Nc'\epsilon_\Delta'\kappa$' ([PM] *114.01) and denotes the product of the cardinal numbers of the classes which are members of κ.

84.8. The arithmetic exponentiation of the class α by the class β is written '$\alpha \exp \beta$' and is so defined that its cardinal number is the cardinal number of α raised to the νth power, where ν is the cardinal number of β. In order to define '$\alpha \exp \beta$', we first define 'Prod' in such a way that $Prod'\kappa = D''\epsilon_\Delta'\kappa$. Thus '$Prod'\kappa$' denotes the class of all classes that are selected classes from κ (66.11, 66.31). If κ is a class of mutually exclusive classes, then the $D''\epsilon_\Delta'\kappa$ will have the same cardinal number as $\epsilon_\Delta'\kappa$, namely the product of the cardinal numbers of members of κ (84.7). If we choose κ as the class $\alpha \downarrow ''\beta$ (65.7), then κ is a class of mutually exclusive classes each having the same number of members as α, and κ has the same number of members as β. Hence $Prod'[\alpha \downarrow ''\beta]$ has μ^ν members, where μ is the cardinal number of α and ν is the cardinal number of β. Hence we may define '$\alpha \exp \beta$' as '$Prod'[\alpha \downarrow ''\beta]$' ([PM] *116.01) and we may define 'μ^ν' as '$\hat{\gamma}(\exists\alpha)(\exists\beta)[[\mu = Nc'\alpha].[\nu = Nc'\beta].[\gamma \text{ sm } [\alpha \exp \beta]]]$' ([PM] *116.02).

85. The definition of relation-numbers by ordinal similarity

85.1. A theory of relation-numbers is developed in [PM] *150–*183 in a way that parallels the theory of cardinal numbers of [PM] *100–*126. This theory of relation-numbers is a generalization and extension of the theory of ordinal numbers developed by G. Cantor (*Mathematische Annalen*, Vol. 46 (1895) pp. 481–512; Vol. 49 (1897) pp. 207–246). Some of the principal concepts and symbols used in [PM] will be briefly described in what follows.

85.2. Two relations are called *similar* (or *ordinally similar*) if there exists a one-to-one correspondence between their fields that preserves the ordering of the fields imposed by the two relations. Thus if the relation P relates x_1 to x_2, and x_2 to x_3, and relates nothing else, and if the relation Q relates y_1 to y_2, and y_2 to y_3, and relates nothing else, we could set up a one-to-one correspondence of the required sort by choosing a one-one relation S (64.33) that relates x_1 to y_1, and x_2 to y_2, and x_3 to y_3. Thus S provides a one-to-one correspondence of the field of P with the field of Q. In saying that S preserves the ordering of the two fields (imposed respectively by P and by Q) we mean that if a bears P to b then the thing to which a is related by S bears Q to the thing to which b is related by S. Hence the members of the field of P are related by P in exactly the same way that their S-correlates in the field of Q are related by Q. More

specifically, P is (ordinally) similar to Q if and only if there is a one-one relation S which has the field of Q for its converse domain and which is such that $P = S|Q|\breve{S}$. This latter condition provides (for all a and b) that a bears P to b if and only if a bears S to something (the S-correlate of a) which bears Q to something (the S-correlate of b) which is borne S by b. We call such an S an *ordinal correlator* of P with Q. The class of ordinal correlators of P with Q is written as 'P $\overline{\text{smor}}$ Q', and we define 'P $\overline{\text{smor}}$ Q' as '$\hat{S}[[S \in 1 \to 1].[C`Q = C`S].[P = S;Q]$' ([PM] *151.01), where '$S;Q$' is an abbreviation for '$S|Q\breve{S}$' (66.27). Just as $\alpha = R``\beta$ in case R is a cardinal correlator of α with β (81.1), so that α is a map or projection of β by R, so also $P = S;Q$ in case S is an ordinal correlator of P with Q, so that P is the map or projection of Q by S (66.27). (See 08.9 with respect to 'C'.)

85.3. Relations P and Q are said to be ordinally (or relationally) similar if there is a relation S which is an ordinal correlator of P with Q, that is, if the class P $\overline{\text{smor}}$ Q is non-empty. Hence the relation of ordinal (or relational) similarity, denoted by 'smor', may be defined by treating 'smor' as an abbreviation for '$\hat{P}\hat{Q}[\exists!P \overline{\text{smor}} Q]$' ([PM] *151.01).

85.4. Just as the cardinal number of a class α is the class of classes similar to α, so also the *relation number* of a relation Q is the class of relations ordinally similar to Q. Thus we can regard Nr'Q (the relation number of Q) as $\hat{P}[P \text{ smor } Q]$ ([PM] *152.1) and NR (the class of relation numbers) as $\hat{P}(\exists Q)\,[P = \text{Nr}`Q]$, that is, as D'Nr ([PM] *152.02).

85.5. In order to describe a special class of relation numbers known as *ordinal numbers* it is necessary to explain what is meant by a *series* and more particularly, a *well-ordered series*, since ordinal numbers are relation numbers of well-ordered series. A series is a relation which is irreflexive (that is, never relates anything to itself, and hence is a subrelation of J, the relation of non-identity (64.12)), transitive (61.7), and connected (64.15). Thus the class Ser of series relations is Rl'$J \cap \text{trans} \cap \text{connex}$ (66.05, [PM] *204.01). A relation P is said to be *well-ordered* if every non-empty subclass of its field has a *P-minimum* member, that is, a member that bears P to some member of the class but is not borne P by any member of the class. The class of well-ordered relations is denoted by 'Bord', where 'Bord' is an abbreviation for '$\hat{P}[\text{Cl ex}`C`P \subset \text{Cl}`\text{min}_P]$'

([PM] *250.01), and where Cl ex'$\alpha = \hat{\beta}[[\beta \subset \alpha].\exists!\beta]$ ([PM] *60.13) and where $\text{min}_P = \hat{x}\hat{\alpha}[x \in [\alpha \cap [C`P - \breve{P}``\alpha]]]$ ([PM] *93.02). The class of well-ordered series is denoted by '$\Omega$', where '$\Omega$' abbreviates 'Ser $\cap$ Bord' ([PM] *250.02). The class of ordinal numbers is denoted by 'NO', where 'NO' abbreviates 'Nr``Ω' ([PM] *251.01). Thus ordinal numbers are relation-numbers of well-ordered series. (See 08.9 with respect to 'C' and 'J'.)

86. Arithmetic operations upon relation-numbers

86.1. The definitions with which we shall be concerned are closely parallel to those of 84. In each case an operation on relations is first constructed; and the result of this operation on relation-numbers is then defined as the relation-number of the result of the operation upon relations. But there are two fundamental differences from 84: (1) Operations upon relations replace operations upon classes. [PM] uses the operation denoted by ';' (66.27) instead of the operation denoted by '``' (63.4). (2) The result of the operation is no longer a class but a relation. Roughly speaking, relations are put one after another (in sums) or interwoven (in products and powers); this entails complications in the definitions. Details will be omitted.

86.2. The *sum* of two relations P and Q (omitting some special cases ([PM] *161 and *181)) is denoted by '$P \ddagger Q$' ([PM] *160.01). It is constructed by adding, so to speak, the relation Q after the relation P. The relation-number of $P \ddagger Q$ is the sum (in the sense of '$+$' below) of the relation-numbers Nr'P and Nr'Q provided that P and Q do not overlap. A sum $P + Q$ is then defined which yields the desired result if P and Q do overlap ([PM] *180.01). If μ and ν are the relation-numbers respectively of P and Q, then $\mu + \nu$ (sum of relation-numbers) is the relation-number Nr'$[P + Q]$. The sum $\Sigma`P$, which is the sum of the relations of the field of $P$ ([PM] *162.01) is the analogue of $\Sigma`\kappa$ (84.4). The sum $\Sigma\text{Nr}`P$ ([PM] *183.01), which is the sum of the relation-numbers of relations of the field of $P$, is the analogue of $\Sigma\text{Nc}`\kappa$ (84.4).

86.3. The product $Q \times P$ of the two relations P and Q is taken to be $\Sigma`[P \downarrow; Q]$ ([PM] *166.01). This is the analogue of $s`[\alpha \downarrow ``\beta]$ (84.5). If μ and ν are the relation-numbers respectively of P and Q, then $\mu \times \nu$ (pro-

duct of relation-numbers) is $\mathrm{Nr}`[P \times Q]$. The product $\Pi`P$ of the relations of the field of P ([PM] *172.01) is the analogue of $\epsilon_0`\kappa$ (84.7). The product $\Pi\mathrm{Nr}`P$ ([PM] *185.01) of the relation-numbers of the relations of the field of P is $\mathrm{Nr}`\Pi`P$ and is the analogue of $\Pi\mathrm{Nc}`\kappa$ (84.7). (See 08.9 with respect to 's'.)

86.4. As for *exponention*, $\mathrm{Prod}`P$([PM] *173.01) is the analogue of $\mathrm{Prod}`\kappa$ (84.8). P exp Q is taken to be $\mathrm{Prod}`[P \downarrow Q]$ ([PM] *176.01) and is the analogue of $\alpha \exp \beta$ (84.8). Finally, if $\mu$ and $\nu$ are the relation-numbers respectively of $P$ and $Q$, then $\mu \exp_{,} \nu$ (*exponentiation of relation-numbers*) is $\mathrm{Nr}`[P \exp Q]$ ([PM] *186).

CHAPTER 9

METAMATHEMATICS

90. The subject of metamathematics

90.1. We have in earlier chapters discussed (logico-mathematical) systems of various sorts. Some systems may be viewed as systems of uninterpreted symbols, others may be viewed as systems of interpreted symbols, and still others, perhaps, may be viewed as systems of concepts (or meanings) in partial or total abstraction from whatever symbols happen to be used to express these concepts. Any discourse about specific systems, or about systems in general, in any or all of the above three senses, might be described as *metamathematical* or *metalogical* discourse, and the language used in such discourse might be called a *metalanguage* (08.1). In what follows we view systems primarily as systems of interpreted or uninterpreted symbols. The formal study of symbols of systems either in their relations to one another (*syntax*) or in their relations to assigned meanings (*semantics*) is called *metamathematics* or *metalogic*.

90.2. To say that a certain sentence (viewed as a string of symbols) is *provable* in a given formal system means that a proof of that sentence exists; that is, that a sequence of sentences exists whose last term is the given sentence and each of whose terms is either an axiom of the system or deducible from preceding terms (or from a class of preceding terms) of the sequence by the rules of deduction (underived rules of inference of the system (01.12)). Provability in a given system is thus a property of sentences (not, e.g., of what they may express) and hence is a metamathematical concept. Since it is definable without reference to meaning, the notion of provability is a syntactical notion rather than a semantical notion (01.13).

90.3. To say that a certain sentence is *true* in a given system refers to an

interpretation of that system, that is, to a relation between the symbols of the system and the entities dealt with by the system (or meanings assigned to symbols of the system). The notion of truth is thus a metamathematical, and specifically a semantical, notion. Sometimes it is possible to define semantical notions in syntactical terms. For example, if S is one of the usual interpreted systems, the notion, *true in* S, can be defined syntactically, that is, simply by reference to symbols and combinations of symbols, without any mention of the objects or meanings denoted or expressed by symbols. The investigation of such possibilities of definition is, however, still considered to be part of semantics rather than of syntax. One of the principal results of this part of metamathematics is the result of Tarski (IX 68), that for any of the usual interpreted systems S, the notion, *true in* S, though syntactically definable, is not definable within S.

90.4. Investigations in syntax revolve chiefly around the problem of consistency. A system is defined to be consistent (specifically Post consistent) when not every sentence of that system is provable (cp. [Ch] 108), and the notion of provability, as we saw above, is a notion of syntax. The question of consistency, and in particular the problem of proving the consistency of a formal system adequate for classical mathematics, is a central problem in the foundations of mathematics. It was in order to solve this problem that metamathematics was founded by David Hilbert (cp. 1085). However, a difficulty appears as soon as this problem is stated. If we hope to prove the consistency of mathematics by a mathematical study of symbols and their properties and relations, our procedure is likely to be circular. For we are assuming the consistency of the mathematics we use in the proof of consistency. In view of this difficulty Hilbert was forced to restrict the mathematical tools which were to be used in his proof of consistency to certain extremely elementary methods whose infallibility is considered beyond question. These methods he called *finitary methods*. Many logicians (Hilbert, Kleene) use the term *metamathematics* to refer only to investigations of syntax (and, to a slight extent, semantics) by such methods. However, we use the word throughout this chapter in the wider sense assumed above (90.1).

90.5. To conclude this brief survey of some main results and problems of metamathematics we may mention the questions concerning the re-

lations between syntactical and semantical notions. For example, let a formula of the first order (Chapter 2) or higher order (Chapter 3) functional calculus be called *valid* if it is true under the usual interpretation of the quantifiers and connectives and under *every* interpretation of the functional letters. The notion of validity is clearly semantical, and it is important to investigate its relation to provability. In this connection Gödel has proved that these notions coincide for the first-order functional calculus in its usual formulation (418*1*) but not for higher-order functional calculi in any formulation (418*3*). In other words, a higher-order functional calculus which contains no invalid formulas among its theorems must fail to contain some valid one, and the same situation would obtain after adding any finite number of axioms.

91. Metamathematical variables and constants

91.1. In order to pursue metamathematics in a rigorous fashion, one needs to distinguish with great care between an expression and its name. Likewise, there must be a careful distinction made between the object-language which is being investigated and the metalanguage in which the investigations are carried out (08.1).

91.2. We must bear in mind that it is not necessary that a rigorous metalanguage make use exclusively of technical symbols. Otherwise we would be involved in a vicious circle: a symbolism of the metalanguage would be necessary to explain rigorously the syntax or semantics of the object-language; then the symbolism of a meta-metalanguage would be necessary to explain rigorously the syntax and semantics of the metalanguage, and so on. On the contrary, the use of a symbolism in the metalanguage is optional. If the metalanguage is (as we presented it) a description and discussion of symbols, the spoken language is ordinarily sufficient to avoid misleading ambiguity. And if a symbolism of the metalanguage is used, this symbolism will almost never be a completely formalized one; it will be comparable to the technical language of the natural sciences. If such a language makes use of symbols, it will be more for the sake of brevity than for the sake of building up a calculus. Hence, our discussion of the metalanguage will have a different character from that of the discussion of the object-language, and we need be concerned only

about acceptable symbols, i.e., symbols which are neither misleading nor excessively complicated.

91.3. Various symbols or expressions of the metalanguage may be used to denote various symbols or expressions of the object-language. One way to form a metalanguage name of an object-language expression is to place single quotation marks around the object-language expression. Thus the object-language symbol written as follows,

A

is denoted by the metalanguage expression written as follows,

'A'.

Another method is the one adopted by Tarski in some parts of his celebrated paper on truth in formalized language (28*514*). Special new expressions are introduced as metalanguage names of object-language symbols, as follows:

Object-language symbol	Metalanguage name of object-language symbol
I	*in* (inclusion sign)
N	*ng* (negation sign)
A	*sm* (sign of logical sum)
Π	*un* (sign of universal quantifier)
x_1	v_1 (first variable)
x_2	v_2 (second variable)
x_3	v_3 (third variable)
.	.
.	

To form the metalanguage name of an object-language expression that consists of two juxtaposed object-language expressions (that have already been assigned metalanguage names), the procedure is as follows: Write

the metalanguage name of the first juxtaposed object-language expression, then write a loop (like an inverted cup), and then write the metalanguage name of the second juxtaposed object-language expression. A similar method can be applied to three or more juxtaposed expressions. For example, the object-language expression written thus,

$$NIx_1x_2$$

is denoted by the metalanguage expression written thus,

$$ng^{\frown}in^{\frown}v_1{}^{\frown}v_2.$$

An expression formed by juxtaposing two or more expressions is sometimes said to be a *concatenation* of such expressions. The concatenation is said to be binary if only two expressions are juxtaposed. Clearly any concatenation can be viewed as reducible to a series of binary concatenations. The loop may be viewed as denoting binary concatenation. The associative law is satisfied by the operation of binary concatenation. (The italic letters in this paragraph are not serving as metavariables but are part of the actual notation being discussed (cp. 08.9). In fact throughout section 91 all displayed expressions are the actual notation being referred to, except, of course, for commas used to separate displayed expressions or for triple dots used to indicate omission of displayed expressions.)

91.4. The variables of the object-language normally range over mathematical and logical objects (sets, numbers, functions, etc.), while the variables of the metalanguage range over expressions of the object-language. The principal kinds of object-language variables are free and bound individual variables (ranging over sets and other unspecified objects in [Q], over numbers and other unspecified objects in [K] and [HB], and over unspecified objects in [Ch] and [HA]), propositional variables, and functional variables for propositional functions. The principal kinds of metalinguistic variables are variables ranging over terms of the object-language, variables ranging over (free and bound) individual variables of the object-language, and variables ranging over formulas of the object-language.

91.5. Some indications on the usage of various authors will now be given.

91.51. In [PM] there is no clear recognition of the difference between metalinguistic variables and object-language variables.

91.52. In the Hilbert school Latin letters are used for variables of the object-language and German letters for the metalanguage. In [HA] the only German letters used are

$$\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \ldots$$

ranging over sentences. [HB] adds to these

$$\mathfrak{a}, \mathfrak{b}, \mathfrak{c}, \ldots$$

ranging over free individual variables,

$$\mathfrak{x}, \mathfrak{y}, \mathfrak{z}, \ldots$$

ranging over bound individual variables, and

$$\mathfrak{r}, \mathfrak{s}, \mathfrak{t}, \ldots$$

ranging over terms.

91.53. [Q] has Latin letters (69) for the object-language and Greek letters for the variables of the metalanguage, namely,

$$\alpha, \beta, \gamma, \ldots$$

ranging over individual variables (75),

$$\varphi, \psi, \chi, \ldots$$

ranging over statements (35), and

$$\xi, \eta, \sigma, \ldots$$

ranging over terms (135).

91.54. [K] uses roman letters for the metalanguage, namely,

$$x, y, z, \ldots$$

ranging over individual variables (146),

$$A, B, C, \ldots$$

ranging over sentences (139), and

$$r, s, t, \ldots$$

ranging over terms. The variables of the object-language are for [K]

$$a, b, c, \ldots$$

ranging over individuals, and

$$\mathscr{A}, \mathscr{B}, \mathscr{C}, \ldots$$

ranging over sentences and functions.

91.55. [Ch] uses Latin italics for the object-language and bold type for the metalanguage, namely, bold roman small letters

$$\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$$

ranging over primitive constants and variables of the object-language (60), and bold roman capitals

$$\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$$

ranging over well-formed formulas of the object-language (60). Use is made in [Ch] also of bold Greek letters, namely, bold Greek small letters

$$\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \ldots$$

ranging over the primitive symbols of the object language (60), and bold Greek capitals

$$\boldsymbol{\Gamma}, \boldsymbol{\Delta}, \ldots$$

ranging over formulas of the object-language.

91.56. All variables in [R] may be considered as variables of the metalanguage (cp. [R] 82 ff.).

92. Designations for classes of symbols and classes of entities

92.1. The symbols in German letters, frequent in Carnap's works, are names for classes of symbols and sometimes for classes of entities. The principal ones are listed below in alphabetic order. These symbols are to be found in Carnap IV 82. The page numbers are indicated.

$\mathfrak{A}$	Finite expression (*Ausdruck*) (17)
$\mathfrak{a}$	Symbol (17)
$\mathfrak{Afu}$	Expressional function (*Ausdrucksfunction*) (191)
$\mathfrak{Ag}$	Expressional framework (*Ausdruckgerüst*) (187)
$\mathfrak{Arg}$	Argument-series (26, 187)

f	Functor-variable (84)
Fu	Functor-expression (84)
fu	Functor (17, 188)
K	Class of expressions (37)
k	Constant (84)
Op	Operator (191)
p	Predicate-variable (195)
pr	Predicate (17, 188)
S	Sentence (*Satzformal*) (25)
s	Sentential variable (84)
sa	Sentential symbol (84)
Sfu	Sentential function (*Satzfunktion*) (191)
Sg	Sentential framework (*Satzgerüst*) (187)
V	Variable-expression (191)
v	Variable (194)
verkn	Junction-symbol (*Verknüpfungszeichen*) (17)
Vf	Sentential junction (201)
vf	Junction-symbol (201)
Z	Numerical expression (*Zahl*) (205)
z	Numerical variable (17)
zfu	Numerical functor (205)
zpr	Numerical predicate (205)
zz	Numerical symbol (Numeral) (*Zahlreichen*) (26)

92.2. The symbols in the preceding list are properly simply abbreviations for the technical terms mentioned. Superscripts are added to mark differences of type and order. But the same symbols are also used, e.g., in axiom schemata, as metalinguistic variables in the sense of 91; subscripts are then added to distinguish different variables of the same kind.

92.3. As in the case of other metalinguistic variables, connectives of the object-language are used by Carnap to join these expressions that serve as metalinguistic variables. This provides an easy way to refer to arbitrary object-language expressions of some special form (cp. 93.4 below). For instance *the disjunction of two sentences* may be expressed by

$$\mathfrak{S}_1 \vee \mathfrak{S}_2.$$

93. Designations for expressions

93.1. In ordinary English there are two ways of forming the name of an expression, illustrated in the following two sentences:
(1) Rose is a noun.
(2) 'Rose' is a noun.
In the first of these ways, called by Carnap the autonymous mode of speech (IV 82), an expression is used as a name of itself. In the second, the name of an expression is formed by putting that expression in quotation marks. The former method is used in metamathematics by all logicians whose usage is mentioned in 91, except for Quine, who favors the second. The present Dictionary agrees with Quine in using the second method (cp. 08). Although the first method is both the simpler and the more usual of the two, its employment requires that certain precautions be observed. The principal of these is that *no variable of the object-language can* without risk of ambiguity or contradiction *serve also as a variable of the metalanguage*, and more generally that no expression of the object-language can function in the metalanguage otherwise than as an autonymous designation for itself. For instance, conjoined with sentence (1) above (in which English is at once object-language and metalanguage) the evident truth that
(3) Rose is a botanical species,
may yield the absurd conclusion that some nouns are botanical species, and this because 'rose' is being used both as a name in the object-language and as an autonymous designation in the metalanguage. To take a formal example, if '*x*' and '*y*' are being used as variables of the object-language and of the metalanguage, and if they are also being used in the latter as autonymous designations (constants) referring to variables of the object-language, we are at a loss as to how to interpret a sentence like
(4) If *x* is a variable, then *x* is called free in *y* if ....
Are we, for example, to take the first '*x*' in this sentence autonymously? If so, the whole sentence becomes synonymous with:
(5) If the twenty-fourth letter of the lower-case alphabet is a variable, then ....
Or are we to take that first '*x*' in (4) as a variable of the metalanguage? In this case, the whole sentence becomes synonymous with:
(6) If anything is a variable, then it is called free in an expression if....

Whenever a variable of the object-language is simultaneously a variable of the metalanguage, confusions of this kind are likely to occur. Similarly confusions arise when any expression of the object-language appears other than autonymously in the metalanguage. The general form of the ambiguity which arises in such cases is reflected in the question: Is the expression to be given the meaning which it has when used non-autonymously in the metalanguage, or should the expression be construed autonymously? Such ambiguity can be avoided in two ways: 1) by prohibiting autonymy altogether, or 2) by prohibiting expressions of the object-language from appearing in the metalanguage in any way other than autonymously. Quine elects the former of these alternatives. Kleene and many others elect the latter.

93.2. In Kleene's usage, despite the autonymous use of expressions, it is impossible for confusion to arise. (The usage of [HA], [HB], and [Ch] is sufficiently similar to that of [K] not to require a separate discussion.) Had Kleene wished the above sentence (4) to be interpreted in the first of the mentioned ways (the autonymous way), he would have left it as it stands. The sentence is then rather pointless because the letter '*x*' is not free in the letter '*y*'; indeed nothing is free in '*y*' but '*y*' itself. The second sense of the sentence (4) above – the interpretation given as (6) (the important non-autonymous sense) – would be rendered in Kleene's notation (91.54) by

(7) If x is a variable, then x is called free in A if....

93.3. Quine's method, on the other hand, permits the use of any style of variable in the metalanguage. The sentence (4) can be given only the non-autonymous interpretation in this notation. Actually, Quine would more likely write (91.53):

(8) If α is a variable, then α is called free in φ if....

But nonetheless (4) is perfectly correct and unambiguous from Quine's point of view. The idea expressed in Kleene's notation by (4) would have to be expressed in Quine's notation (and indeed in the notation of this Dictionary) by:

(9) If '*x*' is a variable, then '*x*' is called free in '*y*' if....

93.4. A difficulty appears in the translation of sentences like:

(10) The result of writing a sentence followed by '&' followed by a sentence, is again a sentence.

A natural way to write it would be:

(11) If A and B are sentences, so is A & B.

Or perhaps

(12) If 'A' and 'B' are sentences, so is 'A & B'.

Actually, neither of these notations is correct unless a special convention is made. Kleene and all authors tabulated above, except Quine, make such a special convention in favor of (11). Evidently 'A' and 'B' in (11) are being used as variables of the metalanguage (hence not autonymously) but are being combined with the help of '&', which is an object-language symbol. This procedure offers no difficulty if we adopt the convention that '&' may be used in the metalanguage to denote a function which when applied to two sentences yields the result of joining those two sentences by '&'. This convention would of course apply also to operators in general in the same way as to '&'.

93.5. Kleene systematically makes this convention, so that in his notation (11) is entirely in order. More specifically, if an uninterrupted expression *E* appearing in the metalanguage is composed partly of expressions of the object-language and partly of variables of the metalanguage, the expression is to be taken as referring to the results of writing the expressions of the object-language referred to by those variables in place of the corresponding metalinguistic variables of *E*.

93.6. Translation (12) of (10) is hard to justify. For in (12) it is unclear how the quotation marks operate. If 'A' and 'B' are expressions of the object-language, the sentence loses its intended generality; if not, why are they in quotes? Nonetheless, an alternative and perfectly precise method of designating expressions employed by Quine is nearer to this approach than to the approach of Kleene. The basic principle of this notation is as follows: If an expression *x* appears in the metalanguage, if the expression begins with "⌜" and ends with "⌝" (*quasi-quotes*, [Q] 33 ff.) and if between these quasi-quotes occurs nothing which is not a (Greek letter) variable of the metalanguage or a constant of the object-language, then *x* is to be taken as referring to the results of writing the expressions of the object-language referred to by the mentioned variables of the meta-

language in place of those respective metalinguistic variables throughout x, and then dropping quasi-quotes. The quasi-quotes may be omitted in the limiting case where x consists of a single Greek letter. In this Dictionary the special quotation marks, which are analogous to the quasi-quotes of Quine (cp. 08.4), are retained even in this limiting case. Furthermore, the values of the metalinguistic variables may, in this Dictionary, be any expressions and not merely expressions of some specified object-language. They may, for example, be expressions of various metalanguages being discussed, as in 91 and 92 above.

94. Substitution

94.1. The substitution of 't' for 'x' in 'A' may be stated explicitly in various ways. [R] (209) writes '{Sub in A: t for x}'; [CF] (94) write the prefix '$[t/x]$' before the 'A'; [Ch] (72) writes '$S^x_t A$'.

94.2. [K] (78) writes '$A(x)$' instead of 'A' and then denotes the results of substituting 't' for free 'x' in 'A' by '$A(t)$'. (The special quotes used here do not appear in Kleene's own metalinguistic notation.) Ambiguity may result, however, unless a certain convention is made. [K] (78) makes the required convention in the following way. If '$A(a)$', '$A(b)$', '$A(c)$',... occur in a certain discourse where 'A' is a metalinguistic variable having formulas of the object-language as values, the *first one to occur*, say, '$A(b)$', is to refer to an arbitrary object-language formula (subject to any limitations imposed by the discourse) in which, unless expressly stipulated, 'b' is not necessarily free. And any such subsequent metalinguistic formula, e.g., '$A(c)$', is to refer to the results of substituting 'c' for (free) 'b' in '$A(b)$' rather than, e.g., the results of substituting 'c' for 'a' in '$A(a)$'.

94.3. With substitution to be performed on several variables, the order of substitution must be determined. If the substitution is indicated by a series of prefixes, under a form such as '$[a/x]\,[b/y]\,A$' (94.1), it is understood that substitutions indicated by prefixes closer to 'A' precede substitutions indicated by prefixes further from 'A'. The notation of [R] (209) is such that if 'x_1',..., 'x_n' are variables and 'A_1',..., 'A_n' are terms, then '{Sub in S: A_1 for x_1, A_2 for x_2,..., A_n for x_n}' refers to the results

of substituting simultaneously 'A_1' for 'x_1', 'A_2' for 'x_2',..., 'A_n' for 'x_n' in 'S'.

94.4. A notation analogous to that of 94.2 is used in connection with substitution for several variables, as '$A(x_1,..., x_n)$'. It is required both in this case and in the case where there is only one variable which, however, occurs several times, that all substitutions are to be performed simultaneously and not successively. A similar notation is used in [HA] (69) and [HB] (88).

95. Theoremhood and derivability

95.1. It was remarked above (90.1, 90.2) that the most important topics of metamathematics center around *derivability* and in particular *theoremhood* (derivability from axioms). For example, under one common definition of consistency, a system is said to be consistent if not all well-formed formulas of the system are theorems (cp. 90.4).

95.2. To indicate that a certain formula is a *theorem* of a certain specified system, the metalinguistic sign '$\vdash$' is used. This sign may be followed either by the formula in question or by a metamathematical (metalinguistic) designation of all theorems of a given form. In the latter case, the expression following '$\vdash$' is called a theorem schema rather than a theorem. The sign '$\vdash$' is frequently called an assertion symbol. For usage in [Q] see 21.7 above.

95.3. In addition to the use of '$\vdash$' discussed above, such notations as

$$A_1,..., A_n \vdash B$$

are used to indicate that the expression on the right is derivable by stated rules from the expressions on the left ([R] 123).

96. Deduction within a system

96.1. Closely related to the concepts of axiom, theorem, and rule of inference (01.12) is the concept of *derivability*. In fact, provability (90.2) may be regarded as a special case of derivability. To say that a formula

'b' is derivable in a formal system from formulas 'a_1', 'a_2',..., is to say that a sequence of formulas exists such that each formula of the sequence is one of 'a_1', 'a_2',..., or is an axiom or is a direct consequence of one or more of the preceding formulas of the sequence by one of the (underived) rules of inference of the formal system, and such that 'b' is the last formula of the sequence. (Derivability from the empty class of formulas may be viewed as equivalent to provability.) Derivability of a formula from one or more other formulas is sometimes expressed by use of a horizontal line placed so that the derivable formula appears below the line and the formulas from which it is derivable appear above the line (cp. [HB] (i 82)). For example, the expressions

$$\frac{\mathrm{A}}{\sim\sim\mathrm{A}}$$

and

$$\frac{\mathrm{A} \quad \mathrm{A}\supset\mathrm{B}}{\mathrm{B}}$$

would indicate respectively that '$\sim\sim$A' is derivable from 'A', and that 'B' is derivable from 'A' and 'A$\supset$B'. These same ideas would be expressed by Curry (cp. [CF] 381) by use of an arrow, for example as follows:

$$\mathrm{A} \rightarrow \sim\sim\mathrm{A}$$

By use of metalinguistic variables a similar kind of notation can be used to express *deduction schemata*. For example, the expression

$$\frac{\mathscr{A} \quad \mathscr{B}}{\mathscr{A}.\mathscr{B}}$$

could be used as a deduction schema indicating that, in some formal system, the conjunction of two formulas is always derivable from the formulas themselves. Here '$\mathscr{A}$' and '$\mathscr{B}$' would be regarded as variables of the metalanguage and would range over formulas of the formal system (cp. 93.4).

96.2. Natural deduction is represented by arranging premisses and conclusion of each deductive step as in [HB] notation of 96.1. In each case it has to be decided from the context whether the deduction leads to an assertion free from any hypothesis or only to an assertion under hypothesis.

96.3. In [F] (14), in order to be more explicit, a vertical line is drawn to the left of all steps of the deduction leading from hypotheses to a conclusion. A special symbol '—' is placed just to the right of this vertical line and immediately below the hypothesis or hypotheses. Various vertical lines correspond to various partial or subordinate proofs ([F] 20 ff.). This notation is due originally to Jaśkowski (5141).

96.4. The L-methods of inferential calculus, used by [K] and by Curry following Gentzen (4422) introduce statements of a special form, called *sequences*, with a special primitive connective, written '$\rightarrow$' by Gentzen and [K] (44). Curry writes '$\Vdash$' ([CF] 316) instead of '$\rightarrow$'. We may interpret, for example, the expression

$$\mathrm{A} \rightarrow \mathrm{B}$$

as

'A' implies 'B'.

Furthermore, the expression:

$$\mathrm{A}, \mathrm{A}', \ldots \rightarrow \mathrm{B}$$

may be interpreted as:

The conjunction of the formulas 'A', 'A'',... implies 'B'.

Also, the expression:

$$\mathrm{A}, \mathrm{A}', \ldots \rightarrow \mathrm{B}, \mathrm{B}', \ldots$$

may be interpreted as:

The conjunction of the formulas 'A', 'A'',... implies the disjunction of the formulas 'B', 'B'',....

The expression, if any, preceding the '$\rightarrow$' is the antecedent of the sequence, and the expression, if any, following the '$\rightarrow$' is the consequent of the sequence. A comma in the antecedent is to be translated as conjunction, and a comma in the consequent as disjunction. When either antecedent or consequent is empty we may interpret, for example, as follows:

$$\rightarrow \mathrm{B}$$

may be interpreted as:

'B' is implied by the conjunction of the empty class of formulas (and hence by any formula, and hence is logically true).

Similarly, the expression:

$$B \rightarrow$$

may be interpreted as:

'B' implies the disjunction of the empty class of formulas (and hence implies every formula, and hence is logically false).

Gentzen and [K] (88) use Greek capitals to refer to series of zero, one, or several formulas separated by commas. Curry ([CF] 317) writes German capitals for such series.

INDEX OF NAMES

INDEX OF SUBJECTS

INDEX OF SYMBOLS

STUDIES IN LOGIC

AND THE FOUNDATIONS OF MATHEMATICS

W. ACKERMANN	*Solvable Cases of the Decision Problem* • *(1968)*
J. W. ADDISON, L. HENKIN, A. TARSKI (Editors)	*The Theory of Models* • *(1966)*
P. B. ANDREWS	*A Transfinite Type Theory with Type Variables* • *(1966)*
Y. BAR-HILLEL (Editor)	*Logic, Methodology and Philosophy of Science* • *(1966)*
S. A. BASRI	*A Deductive Theory of Space and Time* • *(1966)*
E. W. BETH	*The Foundations of Mathematics* • *(1965)*
I. M. BOCHEŃSKI	*Ancient Formal Logic* • *(1968)*
P. BRAFFORT and D. HIRSCHBERG (Editors)	*Computer Programming and Formal Systems* • *(1963)*
J. N. CROSSLEY (Editor)	*Sets, Models and Recursion Theory* • *(1967)*
J. N. CROSSLEY, M. A. E. DUMMETT (Editors)	*Formal Systems and Recursive Functions* • *(1965)*
H. B. CURRY	*Outlines of a Formalist Philosophy of Mathematics* • *(1963)*
H. B. CURRY, R. FEYS†	*Combinatory Logic* • *(1968)*
R. FEYS and F. B. FITCH	*Dictionary of Symbols of Mathematical Logic* • *(1968)*
R. L. GOODSTEIN	*Recursive Analysis* • *(1961)*
L. HENKIN, P. SUPPES, A. TARSKI (Editors)	*The Axiomatic Method, with Special Reference to Geometry and Physics* • *(1959)*
A. HEYTING	*Intuitionism* • *(1966)*
J. HINTIKKA and P. SUPPES (Editors)	*Aspects of Inductive Logic* • *(1966)*
C. R. KARP	*Languages with Expressions of Infinite Length* • *(1964)*
S. C. KLEENE and R. E. VESLEY	*The Foundations of Intuitionistic Mathematics* • *(1965)*
N. E. KOBRINSKII and B. A. TRAKHTENBROT	*Introduction to the Theory of Finite Automata* • *(1965)*
G. KREISEL and J. L. KRIVINE	*Elements of Mathematical Logic* • *(1967)*
K. KURATOWSKI and A. MOSTOWSKI	*Set Theory* • *(1968)*
I. LAKATOS (Editor)	*Problems in the Philosophy of Mathematics* • *(1967)*
I. LAKATOS (Editor)	*The Problem of Inductive Logic* • *(1968)*
I. LAKATOS, A. MUSGRAVE (Editors)	*Problems in the Philosophy of Science* • *(1968)*
E. C. LUSCHEI	*The Logical Systems of Leśniewski* • *(1962)*
STORRS McCALL	*Aristotle's Modal Syllogisms* • *(1963)*
R. M. MARTIN	*Toward a Systematic Pragmatics* • *(1959)*
E. A. MOODY	*Truth and Consequence in Mediaeval Logic* • *(1953)*
A. MOSTOWSKI	*Sentences Undecidable in Formalized Arithmetic* • *(1964)*
N. RESCHER	*Hypothetical Reasoning* • *(1964)*
A. ROBINSON	*Non-standard Analysis* • *(1966)*
A. ROBINSON	*Introduction to Model Theory and to the Metamathematics of Algebra* • *(1965)*
H. RUBIN and J. RUBIN	*Equivalents of the Axiom of Choice* • *(1963)*
H. ARNOLD SCHMIDT et al. (Editors)	*Contributions to Mathematical Logic* • *(1968)*
M. W. SULLIVAN	*Apolcian Logic* • *(1967)*
A. TARSKI, A. MOSTOWSKI, and R. M. ROBINSON	*Undecidable Theories* • *(1968)*

AMSTERDAM P.O. BOX 1491 NORTH HOLLAND PUBLISHING CO. NETHERLANDS